An investigation of artificially-evolved robust and efficient connectionist swimming controllers for a simulated lamprey

Jimmy (Hajime) C. Y. Or

Doctor of Philosophy

Institute of Perception, Action and Behaviour
Division of Informatics
University of Edinburgh

2002

Abstract

This dissertation investigates the evolutionary design of robust and efficient connectionist swimming controllers for a simulated lamprey. Using the neuromechanical lamprey model proposed by Ekeberg [1993] and extending the work of Ijspeert [1998] on evolving lamprey swimming CPGs using genetic algorithms, I investigate the space of possible neural configurations which satisfies the following three properties : 1) Robustness against variation in body parameters, 2) Swimming efficiency, and 3) Robustness against random variation in neural connections. These properties were chosen because they are important to both the real lamprey and its robotic implementation.

After a review of the relevant literatures on lamprey neurophysiology and a detailed account of Ekeberg's and Ijspeert's works, I describe my reimplementation of their model. Using that model, I study the effect of variation of body scale on the performance of Ekeberg's and Ijspeert's controllers. The controllers interact with the mechanical model in different ways to achieve high speed swimming at various scales. To investigate this phenomenon I characterise the behaviour of the body when driven by a sinusoidal-based analytic controller. The resulting performance data reveals that the mechanical model has two or three characteristic resonances which the swimming controllers exploit.

Efficiency is computed as the ratio of forward speed of the model lamprey to the speed of propagation of the propulsive mechanical wave in the body. Techniques are devised to estimate these quantities and to measure efficiency at various swimming speeds. The measurements are incorporated into the fitness function of Ijspeert's original GA and efficient controllers are evolved. Interestingly, the best evolved controller not only is capable of swimming in a similar manner to the real lamprey, but also with the same efficiency (about 0.8). Moreover, it exhibits a wide range of controllable speeds and efficiencies. Curiously, evolving controllers using this method also produces better controllers - ones that are more robust in terms of less change in speed at different body scales - than the controllers evolved using a fitness function which rewards robustness directly.

Four controllers are tested: Ekeberg's biological controller; Ijspeert's controller 2, my evolved hybrid robust controller; and my evolved efficient controller. Comparing

cases of random variation in intrasegmental or intersegmental weights against the corresponding prototype controller allows estimates of robustness to be made. Results indicate that the best evolved efficient controller and the Ekeberg hand-crafted controller perform the best. Of these two controllers, the former is better in terms of its ability to allow the simulated lamprey to swim at a higher speed and efficiency. By comparing the GA with classical optimization techniques and hand-crafting, I show that the population-based GA, although it might not be the most efficient way to solve the problem in its class, is a more appropriate approach for creating a swimming controller for a simulated lamprey which is not only able simply to swim, but is also robust and efficient (two significant ecological properties). This is the key point of the thesis. Through comparative analysis, I go on to find a list of features of intersegmental couplings which may potentially be responsible for the robustness and efficiency of the swimming controllers.

Since the intersegmental couplings which lead to efficient swimming are still imperfectly understood, the evolved controllers developed in this thesis may help neurobiologists to gain a better understanding through comparative analysis of the neural organizations of the artificial and real neural networks. Even if the neural organization of this controller turns out to be just one possible configuration that holds the properties mentioned above, the results from this dissertation can still be applied to control an artificial lamprey in simulation and possibly a robotic one.

Acknowledgements

Many people have helped me to complete this work. It could not have been done without the support of my excellent supervisors John Hallam, David Willshaw and Auke Ijspeert. John assigned me a project that could be accomplished in three years yet was broad enough to allow me to explore areas such as motor control, genetic algorithms, neural networks, computer simulations, etc. He taught me how to write good computer programs and think like a researcher. Furthermore, he cured my phobia of LATEX which I was too scared to try. Auke shared his expertise and experience of evolving lamprey locomotion controllers, saving me a lot of time. Since the date I picked up the project, he has spent an enormous amount of time on me. I gratefully acknowledge their patience in answering countless e-mail messages and making comments on earlier versions of this thesis. I would also like to thank David for checking my first year research proposal and this thesis. I gratefully acknowledge his help with preparing me for the viva. Thanks to my internal examiner, Dr. Bob Fisher, for his hospitality during my staying in Edinburgh. Special thanks to my external examiner Prof. Jean-Arcady Meyer for coming all the way from Paris to examine me.

I would also like to thank those generous individuals who devoted a lot of their time to proofreading my thesis. In no particular order, many thanks to Catherine Dickie and Elspeth Thomas who have read countless drafts of this thesis. Catherine has been of great help. Even at the end of the term, she proofread the final draft of the entire thesis. Without her dedication and determination, I would not be able to submit my thesis at this time of the year. Thanks also to Bridget Hallam for proofreading the earlier version of this thesis and my paper presented at the Simulation for Adaptive Behavior (SAB 2000) conference in Paris.

Special thanks to Robin Cohen for her kind support over the past 12 years. Without her, I would not have made it to this stage.

Thank you Morag Anderson and Frances Fullarton for assigning me a nice study accomodation at the main library. I spent more time in that room than at home.

A number of other people have given me time, help and encouragement. In no particular order I would like to thank Gen Matsumoto, Colin Stirling and Kentaro Miyamoto.

Warmest thanks to Tomoaki Kasuga for all his help over the years.

I am grateful to the staff of the division. In particular, John Berry for helping me to sort out all the computer related problems over the years. He is extremely responsible and kind and does an excellent job as computing officer. Without him, I would not have been able to complete the experiments on schedule. Thanks also to Janice Gailani and Olga Franks for their help on library matters. Thanks to Judith Gorden for processing my application, and to Emma Black and Karen Taylor for secretarial matters. Thanks also to Chris Jowett for answering questions about the Faculty.

Thanks to all the members (both former and current generations) of the Mobile Robot Group for sharing their wisdom, knowledge and working environments with me.

Thanks to Terry Coles, Irene Brown and Jenny Jerji for their special arrangements for a quiet living environment. Thanks also to my flatmates Hua, Frank and William for a friendly and party-free living environment. Thanks to the cleaning ladies for doing an excellent job to provide a nice working and living environment.

Thanks to David Findlay and his staff at the Learning Resource Center. They taught me how to scan pictures beautifully.

Thanks also to all the teachers, staff and students at the University of Tokyo, University of Waterloo, Alberta College and the Canadian Secondary Middle School for their help.

Thanks to Louis Gerstner (CEO of IBM) and Patrick Kilburn of IBM Canada for a free upgrade of my Thinkpad. Thanks also to the T21 for not breaking down. Thanks also to an anonymous individual who donated an IBM desktop computer (with an Athlon 1G Hz processor). The machine has given me many valuable results.

I would like to acknowledge funding from the British ORS Awards, the Canadian Natural Science and Engineering Research Council and the Canadian Space Agency.

Finally, I am deeply grateful to my parents Kent and Jo Jo for supporting me all these years. I am indebted to them.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

To my parents and those who have been supporting me.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 General Overview

Biological motor control has always been a fascinating research area. The question of how the nervous system controls movements is a fundamental one. Findings from research in this area can provide important contributions to both mankind and robotics. For example, understanding how signals from the central nervous system control countless muscles in the body allows the development of more sophisticated prosthetic devices and better control of robots with multiple redundant actuators.

In recent years, there have been advances in understanding animal motor control due to better physiological measurement techniques, higher density micro-electrodes and faster computers for simulations of the neural mechanisms which underlie behaviors. However, due to the complexity of the nervous systems, we are still far from being able to understand completely the neural control of higher vertebrates such as humans. The lamprey has been chosen for study by several neurobiologists because it is relatively easy to analyse: firstly, because while it has a brainstem and spinal cord with all the basic vertebrate features, the number of neurons in each category is an order of magnitude fewer than in other vertebrates, and secondly because its swimming gait is simple. Hence, findings on this *prototype vertebrate* can provide a better understanding of vertebrate motor control.

1.2 Motivation and solution

Over the past 15 years, neurobiologists have achieved a better understanding of the lamprey locomotive networks. However, nobody yet fully understands how the segmental oscillators inside the lamprey central pattern generator (CPG) are coupled. I believe it is a good idea to consider a few important properties of the lamprey and then use them as a guide towards the discovery of features in its neural organization that are related to such properties. This thesis is a first step in this direction.

In this dissertation, I investigate the following three properties which I believe to be essential to the survival of the lamprey:

1. Robustness against variation in body parameters.

2. High swimming efficiency.

3. Robustness against random variation in neural connections.

Given that there are so many neural connections in the CPG, how can I relate these properties to the corresponding features in the neural networks? The non-symbolic AI technique called Genetic Algorithm (GA) may provide the answer. The GA is a type of computer algorithm inspired by ideas from natural evolution. Unlike other engineering approaches, there is no need to specify how to solve the problem. Instead, the designer need only specify the desired features (behavior) of the solutions and let the evolution algorithm find them. As we can use mathematical formulas to define swimming efficiency and possibly robustness in speed against changes in body scales, the GA is justified as a suitable design tool.

Using the neuromechanical lamprey model proposed by Ekeberg [1993] and extending the idea from Ijspeert's 1998 work on evolutions of lamprey swimming CPGs, I used a GA to evolve robust and efficient swimming controllers for the model lamprey. The evolved controllers are compared with the one hand-crafted by Ekeberg. The results indicate that the best evolved controller, the efficient controller, is able to swim at a higher speed and efficiency than the hand-crafted controller. Furthermore, the efficient controller is also more robust in terms of speed against variation in body parameters and neural connections. Hence, I have demonstrated that the GA is a more

appropriate tool than hand-crafting for creating a swimming controller for a simulated lamprey which is not only able simply to swim, but is also robust and efficient (two significant ecological properties). Note that although there is not sufficient evidence to show that the GA is the most efficient design tool, the discussion in Subsection 6.4.1 indicates that it is a more appropriate one than the classical optimization techniques for solving the problem. The reason for this is that the fitness landscape is rugged and contains multiple peaks. Whereas the classical optimization techniques tend to get stuck at the local optimum and the chance of finding a good solution depends heavily on the starting position, the population-based GA, on the other hand, starts with several scattered regions in the search space and looks for the optimum in parallel. Hence it has a much higher chance in finding a good efficient swimming controller than the classical optimization techniques.

The reasons for not using a robotic lamprey are firstly, because such a robot is not available at the moment; secondly, because testing the evolved controllers on an expensive robot is not feasible - any bad controller can easily damage the robot; and thirdly, simulations allow us to study the effect of each parameter on the behavior of the model lamprey independently (this is particularly useful as the experiments are repeatable).

Note that although this work is inspired by Ijspeert's doctoral dissertation, it is not exactly the same. Ijspeert [1998] was concerned with evolving controllers which allow the simulated lamprey of normal body scale to swim like the real one and with a wide range of speeds.[1] He also looked at the issue of controllability (through experiments which tracked a randomly moving target) and the possibility of building controllers for the simulated salamander by extending the lamprey swimming CPG. My dissertation focuses on extending his technique to develop controllers with the capabilities that are important for the survival of both the real and the artificial fish. Throughout this thesis, locomotion controllers have been evolved to allow the model lamprey not only to swim in a similar manner to the real lamprey, but also with the same efficiency. Note that although Ijspeert mentioned several times in his thesis that his controllers are efficient, he was using a different definition of efficiency. According to Ijspeert (personal

[1] Note that a controller which allows the model lamprey to swim at the highest possible speed alone may not be realistic.

communication), efficiency is the ability to produce a large range of frequencies, phase lags and speeds. This definition is not the same as what is commonly used by biologists, who usually define efficiency as the ratio between forward swimming speed and mechanical wave speed. This is the definition used in this dissertation. Experimental results show that the best evolved controller has the same swimming efficiency as the real fish. So far, no one understands the intersegmental couplings which lead to efficient swimming. The evolved controllers developed in this thesis may help neurobiologists to gain a better understanding through comparative analysis of the neural organizations of the artificial and real neural networks. Furthermore, I have investigated the issue of robustness in speed and/or efficiency against changes in body scales and noise in neural connections. These issues are very important for both the real and the artificial lamprey.

As far as I know, except for my own work reported in Or and Hallam [2000], nobody has studied the robustness of the lamprey controllers against changes in body scales. As for robustness against noise in neural connections, although Ferrar et al. [1993] studied the effect of noise on oscillator patterns, it was only at the segmental level. No one has investigated the robustness of the swimming pattern generators against noise in a complete 100-segment controller.

1.2.1 Original contributions

To the best of my knowledge, this dissertation provides original contributions to both biology and robotics in the following ways:

1. More in-depth study of the mechanical lamprey model.

   As I use the mechanical model to evaluate the performance of the evolved locomotion controllers, it is important to know how the model lamprey behaves. Although Ijspeert [1998] corrected some of the mistakes found in the mechanical model proposed by Ekeberg [1993] and plotted the speed at different frequencies and phase lag surfaces, he did not provide an explanation of how the speed changes at different frequencies and phase lags.

   In this dissertation, I perform a more in-depth analysis of the corrected mechan-

ical model using a sinusoidal controller. In addition to investigating how the model responds (in terms of maximum speed change) at different amplitudes, frequencies and phase lags of signals generated by this analytic controller, I study the effect that varying the body parameters has on the model. This has not been done before. Experimental results indicate that the model lamprey may exhibit multiple resonances (Chapter 5).

2. Evolution of robust swimming controllers.

   In Or and Hallam [2000] and Chapter 4 of this thesis, I reported on neuromechanical simulations of lamprey with variable body configurations and controllers. The controllers under investigation included Ekeberg's biological controller, two evolved controllers from Ijspeert [1998][2] and a sinusoidal controller. The performance of these controllers was assessed by measuring the maximum achievable speed. Experimental results show that all four controllers under investigation behaved similarly with respect to robustness against variation in body scales.

3. Evolution of efficient swimming controllers.

   Although Williams [1986] reported on the swimming efficiency of the real lamprey, currently no one knows how the segmental oscillators are coupled to produce efficient swimming for the survival of the species. The topic of intersegmental couplings has been studied for over 15 years but no one yet knows in detail much about the mechanisms responsible for the coordination of segments [Buchanan, 1992, Wadden et al., 1997, Miller and Sigvardt, 2000]. In Chapter 6 I use the Genetic Algorithm as a design tool for evolving efficient swimming controllers. The best evolved controller is capable of swimming with the same efficiency (about 0.8) as the real lamprey. Through comparing the neural organization of this controller with that of the biological ones, my work may shed light on the mechanisms and organization of intersegmental coordination for efficient swimming. As it is very difficult to hand-craft a controller which can achieve high swimming efficiency, this is one of the key results of this thesis. The approach I have used aims to gain a better understanding of biology via synthesis

[2]These two controllers were evolved to maximise the frequency, phase lag and speed ranges.

with evolutionary algorithms.

4. Study the robustness and optimality of the swimming CPGs against stochastic variation in connection weights.

   Ferrar et al. [1993] investigated the robustness of the lamprey segmental network proposed by Buchanan and Grillner [1987]. They observed how the patterns of oscillations such as cycle duration, burst proportion and left-right phase lag change when the network is subjected to noise in its neural connections. No one has yet conducted studies on how neural characteristics and swimming performance change when noise is present in both segmental and intersegmental connections. Chapter 7 investigates and compares the robustness of the Ekeberg biological controller, Controller 2 evolved in Ijspeert [1998], the Hybrid robust rontroller evolved in Or et al. [2002] and the Efficient controller evolved in Chapter 6. As part of the investigation, the optimization of these controllers across a wide range of global and extra excitations is evaluated. The investigation confirms that the evolved controllers are more optimized for speed than the relatively robust biological controller. Through comparative analysis, I found a list of features of intersegmental couplings which may potentially be responsible for the optimization, robustness and efficiency of the swimming controllers. As far as I know, to date no studies have been carried out on these aspects of intersegmental couplings. The findings of this thesis may provide an inspiration to neurobiologists to perform real biological experiments to confirm my hypothesis, hence hopefully leading to a better understanding of intersegmental couplings. Thus, the findings here are another key result of this thesis.

1.2.2 Thesis organization

The thesis is organized as follows:

Chapter 2: As I am interested in motor control in both vertebrates and invertebrates, some of the common principles in this field are mentioned. Then, I move on to explain why the lamprey is chosen as an experimental subject. Some of the commonly used terms are defined, followed by a detailed description of various components re-

lated to lamprey swimming. A comparison of the different types of models proposed for the swimming central pattern generator (CPG) is given followed by a description of Ekeberg's neural and mechanical models for the lamprey. These models are important as they serve as a basis of comparison for the evolved controllers. As the topic of this dissertation is evolving neural network locomotion controllers for the model lamprey in simulation, some of the related artificial intelligence techniques such as neural networks and genetic algorithm are introduced. Finally, Ijspeert's PhD work on evolving lamprey locomotion controllers is reviewed. The reason for presenting his work is that although I implement the programs myself, the GA parameters and corrected parameters for Ekeberg's mechanical model are taken from his thesis. Furthermore, the fitness functions which I develop for evolving robust controllers and efficient controllers are built on top of his functions.

Chapter 3: This chapter gives a description of my reimplementation of Ekeberg's neural and Ijspeert's mechanical model. The post-synaptic normalization scheme used to implement the CPG is described followed by the quantitative analysis of both the segmental oscillator and the entire swimming controller. A new speed calculation algorithm which takes turning into account is presented, along with a description of the procedure for neuromechanical simulation (for generating various surface plots for visualizing how speed/efficiency and the characteristics of the neural wave vary under different excitation combinations from the brainstem). Since I am trying to use the results from neuromechanical simulations as inspirations for real biological experiments, a discussion of the realism of my simulations concludes this chapter.

Chapter 4: In this chapter, various types of robustness are discussed. I then give the reasons for investigating robustness in swimming speed against systematic variation in body parameters as well as random variation in connection weights. Since part of this dissertation involves evaluating the robustness of the controllers under different body scales, I describe the effect of scaling the model lamprey's body parameters. A preliminary study on the robustness of a few existing lamprey swimming controllers is then presented. In this study, neuromechanical simulations of lampreys with variable

body configurations and controllers are conducted. The performance of the controllers is assessed by comparing the maximum achievable speed. Results indicate that the biological controller hand-crafted by Ekeberg [1993] and controller 2 evolved by Ijspeert [1998] using GA behave very differently. The differences in behavior between these two controllers are a motivation for Chapter 5.

Chapter 5: In this chapter, a series of investigations at both behavioral and neural levels are used to explain the differences in behavior between the biological controller and controller 2. First, a forced pendulum is used as an analogy to understand the anomalous speed curve of controller 2. Then, Experiment 4 (reported in the preliminary study described in Chapter 4) is repeated using the new speed calculation algorithm with higher resolutions in the excitation grids. The neural organization of the two controllers are compared and then a study on characterizing the mechanical lamprey model using a sinusoidal controller is conducted. Throughout the investigations I found that the mechanical lamprey model I am using may exhibit multiple resonance. It is this property and the differences in neural organization which cause the two controllers to behave differently.

Chapter 7: Given that swimming efficiency is important to both the real and the robotic lamprey, this chapter describes how to use GA to evolve swimming controllers using a fitness measure that takes swimming efficiency into account. Some of the evolved controllers can not only swim faster than the prototypes but also swim at a higher efficiency. Coincidentally, I discovered that the efficiency of a few of these controllers remains fairly constant despite changes in speed. Also, some of the controllers are more robust in speed against change in body scales than those evolved using a fitness function which takes into account the variation in performance with body scaling which I reported in [Or et al., 2002].

Chapter 8: This chapter investigates the robustness and optimality of the four most interesting central pattern generators. These are the biological controller, controller 2, the hybrid robust controller and the efficient controller. First, I conducted experiments

on their robustness at the excitation combination which corresponds to either the maximum speed or efficiency by randomly varying the segmental connection weights and on some occasions also the intersegmental couplings, through altering noise ranges. I observed that controller 2 is relatively robust against stochastic variation in neural connections, followed by either the biological or efficient controller (depending on the experiments). The hybrid robust controller has the lowest ability to withstand noise. The results may imply that the neural organization of both the efficient controller and hybrid robust controllers are optimized for high swimming speed and efficiency. I then investigated the robustness of the controllers across a wide range of global and extra excitations. Three different measurement methods are used to evaluate the controllers. The results confirm that the evolved controllers are more optimized than the biological controller for producing high swimming speeds at different excitation combinations. In terms of variability (defined as the average of the sum of the absolute difference in relative speed across different runs), the biological and efficient controllers are equally robust across the different global and extra excitation combinations under investigation. The performance of controller 2 is fair while that of the hybrid robust controller is poor. Interestingly, although the swimming performance (in terms of maximum speed and efficiency) of the biological controller is not as good as that of the artificially evolved controllers (it is actually quite good already), it is relatively robust against noise in the neural networks. This suggests that the natural evolutions have evolved a swimming controller which is good enough to survive in the real world. Finally, through comparative analysis, I provided a list of features of some of the intersegmental couplings which are potentially responsible for the optimization, robustness and swimming efficiency of the controllers. Interaction with neurobiologists will be necessary to confirm my hypotheses and findings.

Chapter 9: Finally, this chapter concludes the work that I have accomplished and draws conclusions from my research as a whole. Some directions for future research are suggested.

Chapter 2

Background

In this chapter, I give an overview of studies on lamprey swimming pattern generators. First, I provide findings on common principles of motor control from studies on vertebrates and invertebrates. Then, I explain the reasons for choosing the lamprey as an experimental subject. Some of the commonly used terms are defined. A top-down approach is then used to give more detailed descriptions of various components related to lamprey swimming. A comparison of the different types of models proposed for the swimming central pattern generator is given followed by a brief presentation of Ekeberg's neuromechanical model (the one used in this thesis). Then, I proceed to an introduction of some of the artificial intelligence techniques commonly used to model biological systems. Work by Ijspeert on automating the development of lamprey swimming controllers is described. Finally, the mutual benefit gained through the interactions between areas such as AI, biology and computational neuroscience is emphasized.

2.1 Common principles of motor control in vertebrates and invertebrates

Researchers in comparative neurobiology have laid out some of the common principles which govern the motor control systems of both vertebrates and invertebrates. The ultimate goal of the field is to understand how systems of nerve cells function together

to produce a wide range of behaviors commonly exhibited by all animals. Some of the functional principles that are common to both the vertebrate and invertebrate motor systems are: 1) motor programming, 2) command systems, 3) afferent regulation of pattern generating networks, and 4) the neuromodulation of pattern generating networks. These principles are presented briefly in the subsections which follow. For details, refer to Pearson [1993].

2.1.1 Principle 1: Motor programming

Motor programming is one of of the most powerful ideas in motor control. According to Marsden et al. [1984], the motor program is:

"a set of muscle commands which are structured before a movement begins and which can be sent to the muscles with the correct timing so that the entire sequence is carried out in the absence of peripheral feedback."

Marsden and colleagues suggested that a range of gross motor programs (such as the locomotor generator in a cat's spinal cord) are hardwired even in the immature nervous system. These programms can be performed without feedback. However, some finer motor tasks (such as writing and fastening buttons) still require somatosensory feedback such as vision and touch.

Evidence which supports the principle of motor programming comes from studies on movements and the corresponding neural patterns of deafferentated[1] vertebrates and invertebrates. Of more than 50 such systems studied, almost all of them are capable of generating rhythmic motor acts such as walking and swimming without afferent inputs [Delcomyn, 1980, Grillner, 1981].

The neural networks which generate rhythmic motor patterns are called the Central Pattern Generators (CPGs). They have been the foundation of research in the field of rhythmic motor systems in both vertebrate and invertebrates for more than two decades. So far, two characteristics have emerged from research on CPGs. First, the majority of interconnections among the neurons in most CPGs are inhibitory. Second, the CPGs can be organized in many ways by different combinations of cellular and

[1]Elimination of sensory input to all or a restricted region of the central nervous system (from Delcomyn [1998]).

synaptic processes in the neurons within the CPGs [Getting and Dekin, 1985, Getting, 1986, 1989]. (Refer to Subsection 2.1.4 for more details.)

2.1.2 Principle 2: Command systems

The motor systems in both vertebrates and invertebrates are hierarchically organized into several functional levels. The general view is that although the CPGs can generate neural activities for rhythmic movements, higher level command systems are still required to regulate activities such as the escape responses of the cockroach [Camhi and Tom, 1978, Ritzmann and Pollack, 1990] and postural adjustments of primates and humans [Bizzi et al., 1991]. This concept is included in the neuromechanical model of the lamprey proposed by Ekeberg [1993] which is used in this thesis. The CPG in this model is capable of producing neural waves for the model lamprey to exhibit anguilliform swimming. By varying excitations from the brainstem (the command system) to the premotor neurons on the left and on the right, the swimming speed and direction can be changed accordingly.

There is one open problem commonly of interest to researchers in the field of motor control. For those studying invertebrates, the mechanism controlling how the information encoded in a population of giant interneurons can be transformed into a coordinated pattern of muscular activity inside the legs (the six legs of the cockroach for example) is particularly interesting. This is analogous to the question of how the information encoded in a population of corticospinal neurons can be transformed in the primate arm to produce movement in the desired directions, pursued by those studying vertebrate motor control [Pearson, 1993].

2.1.3 Principle 3: Afferent regulation of pattern generating networks

Afferent feedback is feedback from the external sensory organs to the central nervous system. It is responsible for regulating the production of motor patterns appropriate for rhythmic movements. Although rhythmic movements in an open-loop system can still be evoked after deafferentation, the movements become less precise and the motor

patterns become abnormal. This effect is profound for behaviors which require close interaction with the environment [Pearson, 1993]. The advantage of having afferent feedback is that muscles can be coordinated with ongoing position and force information from the mechanical body to achieve effective and precise movements. For instance, Ekeberg et al. [1995] and Ijspeert [1998] found that feedback from the edge cells is necessary for the model lamprey to cross a speed barrier.[2]

2.1.4 Principle 4: Neuromodulation of pattern generating networks

Recently, advances in the analysis of pattern generation networks led to the discovery that the functional properties of neural circuits can be modified by neuromodulators. Harris-Warrick [1988] (cited by Pearson [1993]) found 20 motor systems in both vertebrates and invertebrates with neuromodulators that are capable of fine-tuning the characteristics of ongoing behavior such as the amplitude, frequency and speed of movements. It has been shown that the neuromodulators in the lamprey system and those in the stomatogastric nervous system (STG) of decapod crustaceans have this capability [Harris-Warrick and Marder, 1991, Pearson, 1993, Cohen and Harris-Warrick, 1984]. For instance:

1. The modulator can alter the functioning of a single network. This in turn allows the network to produce different motor patterns for different behaviors. This is known as a *polymorphic network*. It was proposed by [Getting and Dekin, 1985]. According to [Harris-Warrick and Marder, 1991, Ayers et al., 1983], a single neural network in the lamprey can accommodate swimming through water, burrowing in mud and crawling on a solid surface. Harris-Warrick and Cohen [1985] suggested that serotonin might be responsible for modulating the rhythmic activity from the CPG to allow these adaptive movements.

2. The modulator can switch neurons from one functional circuit to another. This allows a neuron which is normally involved in the production of one motor pattern to participate in the generation of another. For example, in the lobster's

[2] A speed barrier is a region where the speed of the water flow is higher than that of the lamprey. The water flow direction is opposite to the swimming direction of the lamprey. It can be used to simulate the situation where the lamprey is swimming up a river or across a narrow gap.

STG, there are two different networks, namely pyloric network and cardiac network. The former generates rhythm patterns in the most posterior foregut region while the latter generates rhythmic dilations of an anterior foregut region. The ventricular dilator (VD) neuron of the pyloric network is able to leave its original network and fire exclusively with the cardiac sac network [Hopper and Moulins, 1989]. This kind of neuron switching has not been found in the lamprey nervous system.

3. The modulator is able to merge the functioning of different pattern generating networks. The resulting network is able to generate a new kind of emergent motor pattern which could not have been produced by networks working individually. For example, in a lobster's STG, there are two independent networks. One generates cardiac sac rhythm and the other generates gastric mill rhythm. The red pigment concentrating hormone (RPCH) is able to cause the cardiac and gastric patterns to be active in a single rhythm [Dickinson et al., 1990]. This kind of fusioning function has not been found in the lamprey.

These capabilities are useful because there is a limit on the number of neurons the body can contain.

2.2 Reasons for studying the lamprey and organization of its motor system

Although much knowledge has been gained through the study of invertebrates, some of this knowledge is not transferable to vertebrates because of major anatomical differences between the two groups of nervous systems [Grillner et al., 1991a, Hellgren et al., 1992].

In order to understand the neural control of locomotion in vertebrates, it is necessary to understand the central pattern generator networks of neurons. It is known that such a network is embedded within the spinal cord and is the basis for rhythmic movements [Delcomyn, 1998, Williams and Sigvardt, 1995]. To gain a better understanding of how the CPG operates, it is necessary to identify all the essential neurons as well

as the way they are connected. Due to the complexity of the spinal networks and the locomotion of higher vertebrates such as humans, a simpler vertebrate has to be used as an experimental subject.

The lamprey belongs to the most primitive vertebrate group, called *cyclostomes*. According to Grillner [1996], its evolution diverged from the main vertebrate line about 450 million years ago. It is considered by biologists to be a *prototype vertebrate* because it has a brainstem and a spinal cord with all the basic vertebrate features. Yet, the number of neurons in each category is an order of magnitude fewer than in other vertebrates [Grillner et al., 1991a]. Furthermore, even when the lamprey brainstem and spinal cord are maintained *in vitro* for several days, the motor patterns remain undisturbed. This is advantageous from an experimental point of view because it allows detailed analysis of circuitry and synaptic mechanisms. Given these merits, lamprey swimming circuitry has been chosen and studied extensively by neurobiologists who want to understand the circuitries underlying vertebrate locomotion [Grillner, 1996].

Besides its prototypical nervous system, the lamprey has also been chosen for study because when it swims it displays a very simple gait which is easy to analyze. This is useful for robotic researchers who want to learn more about different ways of controlling multiple redundant actuation systems in an efficient way.[3] According to Grillner et al. [1991a], a lamprey consists of approximately 100 segments, each of which has at least four muscles. Hence, a large number of actuators is involved during locomotion. The question of how the lamprey coordinates its actuators in a flexible and fault-tolerant way is very interesting from an engineering point of view.

Like the locomotion of other vertebrates, the lamprey's locomotion is an emergent behavior which arises from the interactions of neural activity, muscle physiology, and the mechanical properties of both the body and the physical system from which the thrust is developed [Williams and Sigvardt, 1995]. The components and interactions involved in lamprey locomotion are shown in Figure 2.1. In the following subsections, I give a description of various components of the lamprey motor system and show how they interact to produce the type of swimming observed in the real lamprey.

[3]The main difficulty in controlling a robot with many degrees of freedom is that the desired motions usually require coordinated timing of different actuators. A signal sent to even a single actuator can have a big influence on the entire motion [Yam and Or, 1998, Ijspeert, 1998].

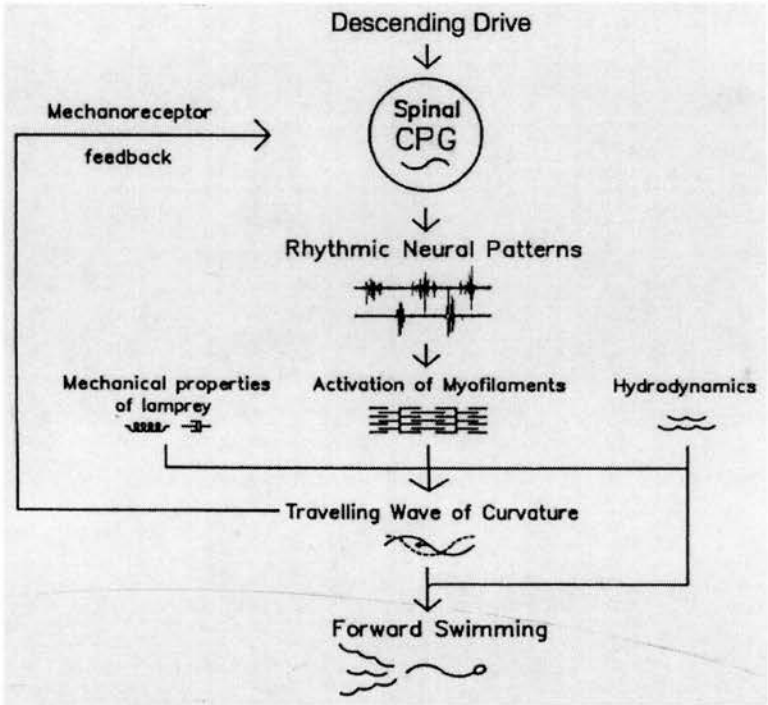


Figure 2.1: Components and interactions involved in lamprey swimming. The descending drive from the brainstem excites the swimming CPG which produces neural activity to allow alternating muscle contractions at each of the 100 segments. The muscular wave and the corresponding mechanical wave then propagate along the lamprey body from head to tail. Anguilliform swimming then emerges. Through the interaction with water, the lamprey is able to swim forward. When the lamprey swims, mechanoreceptors (edge cells) provide feedback to the CPG. (Reproduced from Williams and Sigvardt [1995] with permission.)

2.2.1 Characteristics of the neural wave generated by the swimming CPG

One of the main difficulties encountered by newcomers to the field of lamprey research is terminology. Different researchers use different terms. Sometimes they even use the same term but mean different things by it. In this section, I give an explanation of some of the key ideas based on definitions commonly used by biologists. An understanding of these ideas will be necessary for the rest of this thesis.

Lag and Phase lag

When a lamprey swims, neurons within each body segment generate bursts of alternating muscular activity. In addition, there is a *lag* (also called *intersegmental lag*) between burst onsets from segment to segment. When the lamprey is swimming forward, the lag is from rostral (head) to caudal (tail). However, when it is swimming backward, the lag is in the opposite direction (Figure 2.2).

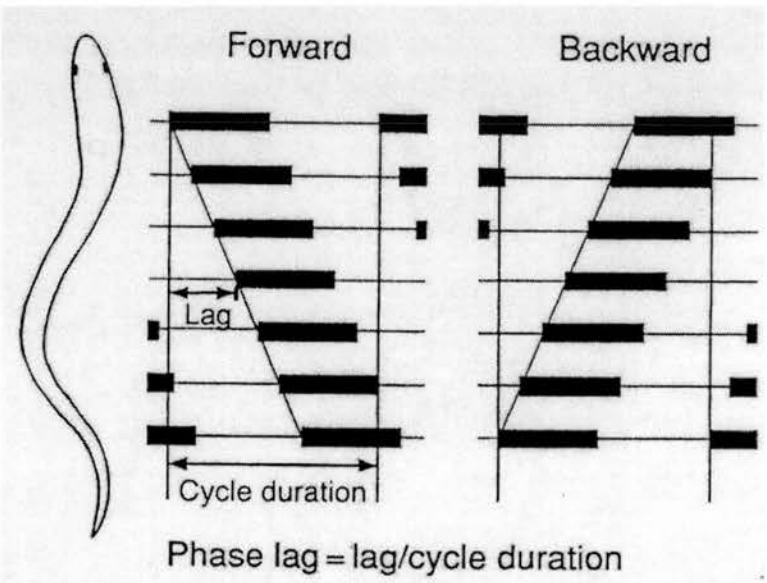


Figure 2.2: Muscular activity during forward and backward swimming in an intact lamprey. The lamprey has an undulatory body shape (*left*). During forward swimming, the undulatory wave is propagated from head to tail. There is a caudal delay between the onsets of bursts from one segment to the other (*middle*). Motor patterns during backward swimming (*right*). Note that the phase coupling is reversed. (Reproduced from Grillner et al. [1995] with permission.)

Note that the lag is always a constant proportion of the cycle duration. Such a relationship is called *phase coupling*. Mathematically, it is defined as:

$$lag = k \cdot cycle\_duration$$

where $k = 1\%$ in the case of the lamprey. Since there are approximately 100 segments, the total phase lag from head to tail is approximately 100% (i.e. one wavelength). The constant k is called *phase lag* or *intersegmental phase lag*. It is usually expressed as a

percentage of the cycle duration.

Cycle duration

Currently, there are several different definitions of *cycle duration*. For example:

1. The time between midpoints of successive bursts of muscular activity in a segment [Wallén et al., 1984].
2. Burst duration plus the interburst interval in a segment [Grillner, 1974].
3. The time it takes for a neural wave to travel from head to tail.
4. The time between the onset of consecutive pulses from the left motoneurons.

The last definition is the one used in this thesis.

Relationships between lag, phase lag and swimming speed

The propagation of a caudally directed muscular wave pushes the lamprey forward through the water. As the frequency of alternating muscle contraction increases (somewhere between 0.25 to 10 Hz), the swimming speed increases [Grillner et al., 1995]. When the undulatory muscular wave is propagated rostrally (towards the head), the lamprey swims backward. Note that the faster the lamprey swims, the faster the mechanical wave propagates down the body.

2.2.2 Characteristics of the lamprey body and their relation to swimming

A lamprey has a smooth, elongated body with scaleless skin, similar to an eel's. On the surface there are anal fins and two dorsal fins. There is no joint or limb but a continuous notochord with soft vertebral arches. When it swims, the lamprey usually keeps the dorsal side up. By propagating a traveling undulation of neural activity from head to tail, a lamprey is able to generate a traveling mechanical wave to propel itself forward through the water.

The gait which a lamprey exhibits during swimming is called *anguilliform* (Figure 2.3). It is characterized by the three special features listed below:

1. There is a monotonic increase in amplitude of lateral movements from head to tail.

2. The traveling undulation propagates along the entire body from head to tail. Note that in the case of the lamprey, there is approximately one wavelength of curvature along the body at any time. This allows the lateral components of the thrust to cancel each other (see arrows in Figure 2.3) while leaving the forward components of the reactive force on the lamprey body to propel the fish forward [Williams, 1986].

3. There are substantial movements of the head.

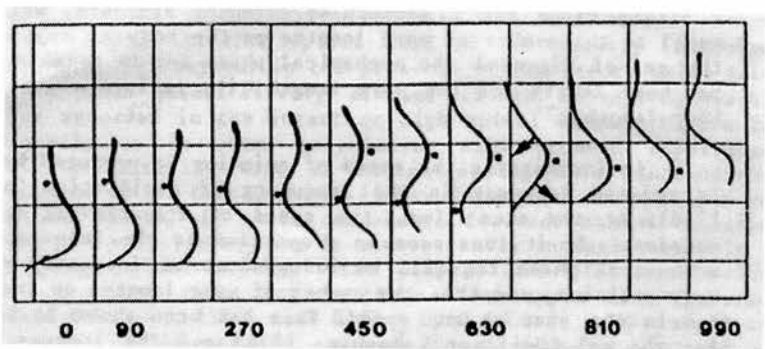


Figure 2.3: Anguilliform swimming of an eel. Grid spacing is 1 inch. Arrows show the direction (but not magnitude) of the force exerted on water. The numbers show time in msec. (Reproduced from Williams [1986] with permission.)

Other main types of swimming modes include carangiform, thunniform (a modified form of carangiform) and ostraciform, etc. A carangiform swimmer swims with body undulations at the last third of its body,[4] and generally speaking, it swims faster than an anguilliform swimmer . However, the latter is better at turning due to the flexibility in its entire body. Among all the swimming modes, the thunniform mode is the most efficient. Thunniform swimmers such as sharks use their extremely stiff caudal peduncle to generate thrust. Finally, there are ostraciiform swimmers. Except for their

[4]Its sibling, sub-carangiform swimmer, swims with body undulations at the last half of its body.

oscillating stiff caudal fin, the rest of their bodies are rigid. This kind of swimming mode is the most inefficient [Sfakiotakis et al., 1999].

2.2.3 Fictive swimming

In order to conduct detailed studies of the neural networks responsible for swimming, Cohen and Wallén [1980] studied *fictive swimming* (see below) induced in an *in vitro* preparation of the lamprey spinal cord. Since there have been reports that even a small movement can entrain the locomotor rhythm of the dogfish [Grillner, 1974], it is very important to establish that all movement-related feedback to the spinal cord has been eliminated. This can be achieved by trimming away all the skin and muscle tissue surrounding the spinal cord and notochord then exposing the spinal cord through the removal of the meningeal tissue. This procedure causes most of the dorsal root connections to the spinal cord to be disrupted. Thus, there is no movement-related sensory feedback to the spinal cord. To eliminate visual feedback, the lamprey is decapitated.

It is known that rhythmic bursting activity in the isolated spinal cord can be induced by adding L-DOPA [Poon, 1980] or the amino acid D-glutamate [Cohen and Wallén, 1980] to the bathing solution. Such bursting activity resembles the patterns of muscle activity observed when the lamprey swims [Cohen and Wallén, 1980, Poon, 1980]. This efferent activity in an isolated or immobilized preparation, which would have caused swimming in an intact lamprey, is called *fictive swimming*.

It is interesting to note that after the application of D-glutamate, rhythmic bursting starts a few seconds later. In some cases, stable rhythmic activity can be maintained for several hours. However, once the drug is removed, the activity stops within a minute. When D-glutamate is applied again, bursting activity reappears. According to Cohen and Wallén [1980], higher drug concentrations can increase the bursting frequency. In addition, there is a species difference in readiness to respond with coordinated motor activity to a non-specific biochemical stimulus. For example, both *in vitro* preparations of the river and silver lamprey produced regular rhythmic activity after the addition of amino acid to the bath. However, the *in vitro* preparation of *Lampetra fluviatilis* did not respond to the drug [Cohen and Wallén, 1980]. Besides chemical stimulation, electri-

cal stimulation at the rostral end of the spinal cord can also be used to induce bursting activity. However, the activities are at very low rates and they last for only about five to ten cycles [Cohen, 1988].

Intrasegmental and intersegmental coordination

Cohen and Wallén [1980] recorded ventral root bursting activity from lamprey spinal cord preparations *in vitro*. They found that there is a clear symmetrical alternation of activity between the two sides of each segment. Moreover, there is a linear relationship between burst duration and cycle duration as well as a relation between intersegmental lag and cycle duration. Other researchers such as [Grillner, 1974, Grillner et al., 1976, Cohen and Wallén, 1980] found similar bursting characteristics in other fish such as the eel (anguilliform swimmer), dogfish, trout and dace. Thus, the lamprey spinal cord represents the neuronal correlate of fish swimming, a basic form of locomotive behavior [Cohen and Wallén, 1980].

Comparison of neural activity in the lamprey spinal cord in vitro, in intact and spinal preparations

Wallén et al. [1984] compared the patterns of muscular activity during swimming in the intact and spinal lamprey[5] with the patterns of ventral root activity in the *in vitro* preparation of the lamprey spinal cord. A summary of the results is as follows:

- **Patterns of activity**
  - The rhythmic activity of all three preparations share common characteristics such as alternating patterns on each side of a segment as well as delays in burst activity between segments. Moreover, they appear to be similar to each other even when they change at different swimming speeds.
- **Timing within a segment**
  - By comparing the burst and cycle durations at different swimming speeds, the burst duration is found to be proportional to cycle duration in all three preparations.

[5]Lamprey with spinal cord transected behind the gills. The wound was closed with sutures.

  - Using the ratio of mean burst duration to mean cycle duration, the burst proportion is found to be almost constant (around 0.4) for all cycle durations.

- **Intersegmental coordination**

  - In all preparations, there is a lag in burst activation measured at different recording positions. This lag increases in proportion to cycle duration. Wallén et al. [1984] defined the term "phase lag per segment" to be the phase lag between two recording positions divided by the number of segments separating them. The phase lag per segment is independent of recording position. Furthermore, all three different types of preparations under study displayed the same principle. Even now, nobody knows exactly what is the underlying mechanism for this constant phase lag. It cannot be due to conduction or synaptic delays because these delays are constant. Some scientists have turned to mathematical modeling for answers (see Subsection 2.3.3).

Other findings on the lamprey motor patterns include the following [Wallén et al., 1984]:

- After spinal transection, there is no spontaneous rhythmic activity in the spinal cord. However, when stimulated mechanically, the lamprey can swim continuously for several minutes.

- During undisturbed symmetrical swimming, movement-related feedback does not have any effect on the burst proportion or intersegmental phase lag.

- Different modes of activation are required to initiate swimming in different preparations. An intact lamprey requires a descending command, spinal preparation requires sensory input to the spinal cord while an *in vitro* preparation requires amino acid in the bathing solution.

- The *in vitro* preparation is important because it allows a much more detailed analysis of neuronal locomotion circuitry which cannot be achieved in the intact preparation.

2.2.4 Organization of musculature in the lamprey body

The organization of lamprey muscular structure shares some of the characteristics found in other fish [Grillner et al., 1988]. These characteristics are given below:

- The muscle fibers have a longitudinal orientation.
- There exists segmental organization, with the number of segments equal to the number of vertebrae.[6]
- During locomotion, activation of muscle fibers proceeds from segment to segment. The excitation of these successive muscular segments leads to the propagating locomotor wave.
- The swimming speed can be increased by either mechanical stimulus (such as pinching the fish at the tail) or from continuous descending signals from the midbrain.
- The strength of stimulation and the number of neurons activated in the brainstem is correlated with the frequency of swimming.
- There are two main types of muscle fiber. 1) Red muscle fibers are slow fibers responsible for normal swimming and 2) white muscle fibers are recruited only during fast swimming. Together, they form a two-geared system for the fish [Wilkie, 1977]. Note that in the Ekeberg [1993] model, there is only one type of muscle.

2.2.5 The lamprey brainstem

According to Grillner et al. [1991b], the lamprey brainstem consists of four different reticular nuclei. The most rostral nucleus is located in the mesencephalon (MRN) and the rest of the nuclei are in the rhombencephalon: the anterior (ARRN), the middle (MRRN) and the posterior (PRRN). Some of these reticulospinal neurons are active

[6]Note that in the lamprey model proposed by Ekeberg [1993] and subsequently used by Ijspeert and myself, the neural networks have 100 segments and the model lamprey has 10 mechanical links, each of which represents 10 neural segments (Subsection 2.5.2).

tonically while others receive feedback from the spinal cord that results in phasic modulation. The glutamatergic reticulospinal MRRN and PRRN neurons project to the spinal cord and are responsible for the initiation of locomotion. These neurons can be activated from the rostral brainstem structures and other sensory stimuli. Upon activation, these reticulospinal cells in turn activate the AMPA/kainate- and NMDA-type glutamate receptors located on the spinal cells of the locomotion CPGs to initiate locomotion. Note that according to Grillner et al. [1991a], one of the main reasons for considering the lamprey to be a prototype vertebrate is that it has a brainstem and a spinal cord with all the basic vertebrate features (such as telncephalon, diencephalon with basal ganglia, mesencephalon and rhombencephalon, different cranial nerves and descending reticulospinal pathways).

2.2.6 The lamprey spinal cord

The lamprey is a member of the primitive vertebrate class named *agnatha* [Wallén and Willams, 1984]. Its spinal cord is on average about 0.3 mm thick and 1 mm wide. It consists of about 100 segments, each of which has a ventral root that the motoneurons pass through to project to the muscles (see Figure 2.4). Along the lateral margin of the spinal cord, there are stretch receptor neurons called *edge cells* (EC) which sense the lateral movements of the body [Grillner and Wallén, 1984].

Typically, an *in vitro* preparation can survive for several days because of the absence of internal blood vessels. It has been shown that perfusing the *in vitro* spinal cord preparation with drugs that mimic the action of excitatory amino acid transmitters (such as the glutamate receptor agonists), can lead to *fictive locomotion* that would have resulted in swimming in an intact lamprey. It is interesting that the same patterns of neural activity as those produced during swimming in the intact lamprey can be produced by the *in vitro* preparation [Poon, 1980, Cohen and Wallén, 1980].

It is known that the superfusion of the preparation by glutamate-receptor agonists leads to the activation of glutamate receptors. Low frequency (from 0.1 to 3 Hz) alternating burst patterns can be elicited by the activation of NMDA receptors. Higher burst patterns (from 1 to 8-10 Hz) can be elicited by the activation of AMPA/kainate receptors [Grillner, 1981, Brodin et al., 1987, Grillner et al., 1995]. By activating these

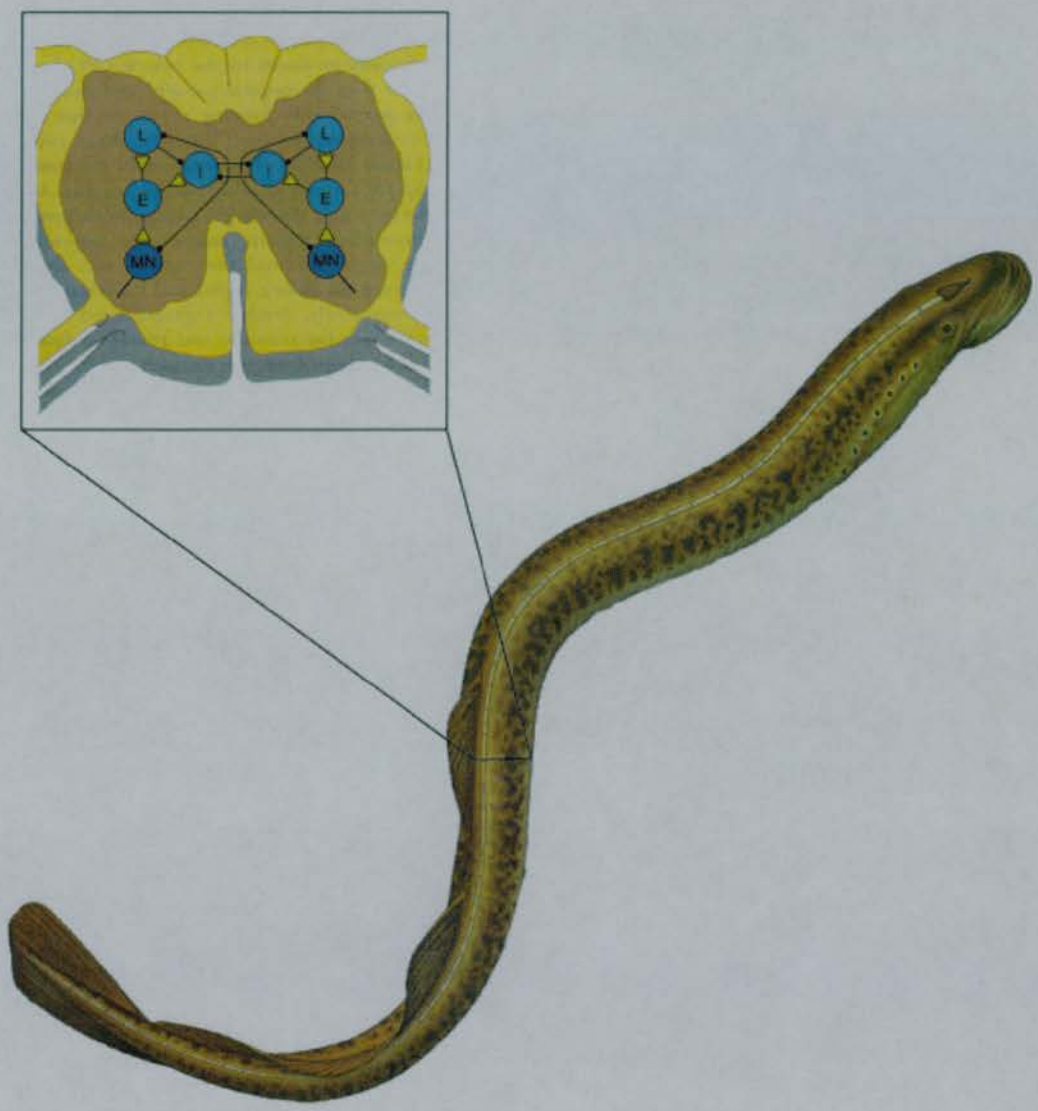


Figure 2.4: A sea lamprey and a schematic view of a segment of its spinal cord. A segmental oscillating circuit responsible for the control of the muscles in a body segment is located inside the spinal cord. On each side of the cord, there are projections from the motoneurons to the muscles through the ventral roots. L, I and E represent the lateral, inhibitory and excitatory interneurons respectively. MN represents the motoneurons. (Modified with permission from Grillner [1996] and Delcomyn [1998].)

different types of glutamate receptors all together, it is possible to generate a frequency range of fictive locomotion which corresponds to that of natural swimming.

2.2.7 The swimming central pattern generators (CPGs)

As mentioned in the previous section, the circuitry which produces oscillatory neural activity for swimming is located in the spinal cord. Knowledge of this pattern generator is gained by research on *fictive swimming* and computer simulations [Cohen and Wallén, 1980, Grillner and Wallén, 1984, Ekeberg et al., 1991, Wallén et al., 1992]. Since an isolated spinal cord of even two segments can still produce oscillations when subjected to an excitatory bath, the swimming CPG is considered to be composed of an interconnection of these local segmental oscillators [Cohen and Wallén, 1980, Grillner et al., 1983].

It is interesting to note that the lamprey CPG as well as the model proposed by Ekeberg [1993] shares some of the characteristics found in other vertebrates [Grillner, 1985, Cohen, 1988, Getting, 1988, Kleinfeld and Sompolinsky, 1989, Grillner et al., 1995, Ijspeert, 1998]. For example:

1. Without sensory feedback from the muscles, the CPG can still produce rhythmic oscillatory bursting patterns.

2. Depending on the inputs, the CPG can produce different patterns of rhythmic behavior. In other words, the same set of neurons can be involved in the generation of different patterns. For instance, during forward swimming, the brainstem sends the same amount of excitation to the neurons on both sides of the CPG. If more excitation is given to the neurons on one side, then the lamprey will turn to that direction.

3. The level of activity of the CPGs is controlled by reticulospinal neurons which project from the lower brainstem to the spinal cord. The higher the level of activity in these neurons, the faster the animal moves.

4. The output from the CPGs can be modulated by external stimuli such as sensory feedback from higher centers or proprioceptors, the sensory receptors (found

mainly in muscles, tendons and joints) that detect the motion or position of the body by responding to stimuli arising from within the body.

5. Most locomotion CPGs are considered to be composed of coupled oscillators or sub-CPGs. For example, the lamprey swimming CPG is thought to consist of 100 copies of the same segmental oscillating network.

Note that one very special property of the lamprey swimming CPG is that it maintains a constant phase lag of 1% between neighboring segments regardless of change in oscillation frequency [Grillner et al., 1988]. The reason I consider the constant phase lag to be special is as following: Supposing there are two coupled oscillators, each of which oscillates at a common frequency of 1 Hz. If there is a time delay of 100 msec between the two oscillators, the phase lag will be 10%. If we double the frequency while keeping the delay constant, the phase lag becomes 20%. Hence, increasing the frequency increases the phase lag. In the case of the lamprey, the phase lag remains constant even when the oscillating frequency changes. Thus, the lamprey CPG is not built by using simple delay.

Note that this property of constant phase lag appears both in intact and isolated spinal preparations [Wallén and Willams, 1984]. So far, several models of lamprey swimming CPGs have been proposed to further understand this feature.

2.3 Models for the lamprey swimming central pattern generators

With advances in neuroscience, biology and biomedical engineering, we are gaining a better understanding of the nervous system. Unfortunately, we still do not have enough information on how different parts of the nervous system (especially in higher vertebrates such as humans) interact to produce desired motor outputs. At present, nobody knows how to decode physiological signals to find out their meanings. In order to gain better understanding of these areas, many neurobiologists have turned to computer modeling to test their hypotheses. In this way they can investigate the behavior of the models under different types of test conditions. This approach leads to the field

called computational neuroscience (Arbib [1995] provides a good discussion/review of neural modeling). Some neurobiologists believe that in order to understand complex nervous systems, it is necessary to gain a better understanding of simpler ones first. Researchers such as Cohen, Grillner, Williams, Ekeberg and Buchanan decided to study the lamprey motor system because of its simplicity and its similarities to that of higher vertebrates (as we have seen in Section 2.2). So far, several models of the lamprey central pattern generator have been proposed. These models are divided into three main groups according to complexity: detailed biophysical models, connectionist models and mathematical models of chains of coupled oscillators.

2.3.1 Biophysical models

In order to investigate the cellular properties of the neurons of the swimming CPG, researchers working on biophysical models incorporate as much detail as possible in their models. It is known that lateral interneurons (LIN), crossed caudal interneurons (CCIN) and excitatory interneurons (EIN) all have membrane potentials which oscillate in phase with fictive swimming [Buchanan, 1982]. According to Rovainen [1974], "there are approximately 45 CCINs and 20-40 EINs per hemisegment and 500-100 LINs in the entire spinal cord, restricted to the rostral half (about 50 segments) of the spinal cord". Buchanan and Grillner [1987] hypothesized that the segmental oscillatory network of a lamprey is composed of these types of neurons. Later, Grillner et al. [1988] made a computer simulation of this network. They used three simulated neurons to represent each of these three neuron types on each side of the spinal cord. The experimental results were encouraging. By adjusting the cellular properties and synaptic strengths parameters, Grillner and his colleagues were able to make the model produce an oscillating pattern similar to that found in lamprey swimming. After the success of the initial experiment, they incorporated more details of membrane and synaptic properties into their original model [Grillner et al., 1991a, Ekeberg et al., 1991, Wallén et al., 1992]. Each neuron in the new model is controlled by approximately 60 parameters to allow the shape of the action potential to look like the real one. Furthermore, the synaptic strengths of the neurons were adjusted to match the activities recorded in the real lamprey spinal cord. Although this model is more realistic than the original

one, it is computationally very expensive.

Given the complexity of the models, investigations were at the segmental level. Although research in this area has provided a better understanding of the burst terminating mechanisms, until recently it was difficult to extend this to intersegmental couplings. With computers getting more powerful, Wadden et al. [1997] pushed the upper limit of computer technology by simulating and investigating intersegmental couplings of networks without segmental boundaries (about 60 segments). This involves 2,400 neurons and 700,000 synapses according to [Lansner et al., 1997]. Although his model behaves similarly to the real lamprey in terms of constant phase lag across the CPG and change in local frequency according to local excitation such as those observed in [Matsushima and Grillner, 1992], the independence between oscillation frequency and phase lag is not as good as its biological counterpart.

The main problem with biophysical models is their complexity. Although computers are becoming increasingly more powerful, they are still not powerful enough to simulate in detail the neural networks of higher vertebrates. Furthermore, even with advances in physiological measurement techniques, some neural parameters still cannot be measured. In order to be biologically compatible, the values of these parameters have to be estimated manually as Grillner et al. [1991a] did. This means that the simulations can become less reliable than the real ones.

2.3.2 Connectionist models

In terms of complexity, connectionist models are simpler than biophysical models but more complex than mathematical models. This approach to modeling the CPG has the advantage that it allows one to focus on the interconnections (such as synaptic weights and delays) between groups of functionally similar neurons and their effects without worrying too much about the details of all the neuron parameters (such as the membrane activity of the neurons). Instead of computing spiking action potentials as the biophysical models do, the mean firing frequency is computed. Hence, computational complexity is greatly reduced. This approach is more concerned with the implementation aspect of intersegmental couplings while the mathematical models of chains of discrete coupled oscillators approach is more concerned with explaining the constant

phase lag of 1% as observed in the lamprey.

Instead of incorporating more biophysical details into the original model developed with Grillner et al. [1991a], Buchanan [1992] used a connectionist model with fewer parameters. In his model each neuron has one variable to represent its membrane potential or the frequency of action potentials. The activation level of each neuron is given by the net input of all other neurons (with excitatory or inhibitory connections) together with an external input. If the net sum of these units is positive, the activation level is used to represent the neuron output. Other parameters for a neuron are threshold, synaptic weight and a time constant. By adjusting these parameters, Buchanan was able to produce oscillating activity like that of their biological counterparts. The result of this experiment is far reaching: it demonstrates that connectivity alone is sufficient to produce oscillatory activity similar to the more complex model developed by Grillner (Subsection 2.3.1). Several connectionist models of the lamprey's swimming CPGs have since been proposed (see [Buchanan, 1992, Williams, 1992a, Ekeberg, 1993, Ferrar et al., 1993, Ekeberg et al., 1995, Jung et al., 1996] for example).

Results from [Buchanan, 1992, Williams, 1992a, Ekeberg, 1993, Jung et al., 1996] prove that segmental connectivity alone can produce oscillations with frequency varying with external excitation. This is similar to the results from physiological experiments. As for intersegmental couplings, findings from [Buchanan, 1992, Williams, 1992a] show that constant intersegmental phase lags can be obtained when there are asymmetric couplings between the segmental oscillators. Like in the real lamprey CPG, if the synaptic weights at the front are stronger than those at the tail, the neural activity travels from the head to the tail. Moreover, increasing tonic excitation to the EIN neurons increases the frequency of oscillation. Independence between the oscillation frequency and phase lag can be achieved. However, if excitation to all the cells is increased, the frequency of oscillation and phase lag is increased. This is contrary to the behavior observed in real biological experiments [Buchanan, 1992, Williams, 1992a].

Ekeberg [1993] proposed a model for a lamprey swimming in a 2D environment. This model consists of a neural simulation of the swimming CPG (consisting of 100 segments) as well as a mechanical simulation of the lamprey body. (More details are

presented in Section 2.4.) The reason for having the mechanical model is to test the neural model's ability to drive an artificial lamprey with body characteristics similar to the real one (but 1/10 of the segments). This is the model which inspired Ijspeert's and my own research. In the neural model, a leaky integrator with frequency adaption is used. Experimental results showed that the neural model is able to produce neural waves with an intersegmental phase lag which is both constant across the spinal cord and independent of the oscillation frequency (recall that the biophysical model proposed by Wadden et al. [1997] cannot produce a neural wave with phase lag independent of frequency). Together with the mechanical model, the artificial lamprey is able to produce swimming patterns similar to those observed in the real lamprey (but with simpler networks and a stationary water medium). Furthermore, the lamprey is able to vary its swimming speed and turn according to excitations from the brainstem. Following the success of this work, Ekeberg et al. [1995] improved his models to allow swimming in 3D. The neural model called *cross oscillatory networks* allows independent control of four different muscles in each segment. With proper timing of muscle contractions, each body segment can bend in any direction to produce yaw, pitch and roll movements. Through the addition of stretch sensitive edge cells into the neural model, Ekeberg was able to study the effect of sensory feedback on swimming behavior. Experimental results showed that sensory feedback is useful in allowing the lamprey to cross a simulated speed barrier. However, the 3D lamprey model has a strong tendency to roll. The problem is fixed by incorporating vestibular organs to the neural model [Orlovsky et al., 1992, Deliagina et al., 1992a,b].

Since the simulated model lamprey is able to swim like a real one, Ekeberg's connectionist models are shown to be biologically plausible and possibly suitable for controlling a real robotic lamprey.

2.3.3 Mathematical models of chains of discrete coupled oscillators

Although the models proposed by Ekeberg [1993] and Wadden et al. [1997] can produce neural waves with constant phase lags across the spinal cord, they cannot offer an explanation of how this is achieved. [Cohen et al., 1982, Rand et al., 1988, Williams

et al., 1990, Kopell et al., 1991, Cohen et al., 1992, Williams, 1992b, Williams and Sigvardt, 1995, Sigvardt and Williams, 1996, Marder et al., 1997] turned to abstract mathematical modeling. The advantages of this approach over the previous two is that the details of cellular properties as well as the connectivities of the neurons are not considered. Thus, formal analysis of the effects of the couplings between the oscillators can be performed. However, this leads to difficulty in relating the findings to the biological reality. For the models to work, two underlying assumptions are needed:

1. A non-linear oscillator (or a pair of oscillators) is used to represent a segment. When isolated, such an oscillator exhibits an asymptotically stable periodic oscillation called the *limit cycle* γ. Furthermore, each oscillator consists of only one dependent variable called the *phase* $\theta(t)$. As the phase increases, an associated dynamic variable such as the membrane potential for example, $x(t)$, rises and falls periodically. This in turn corresponds to the on-off bursting pattern of the ventral root of a spinal cord [Cohen et al., 1982]. (See Figure 2.5 for an illustration.)

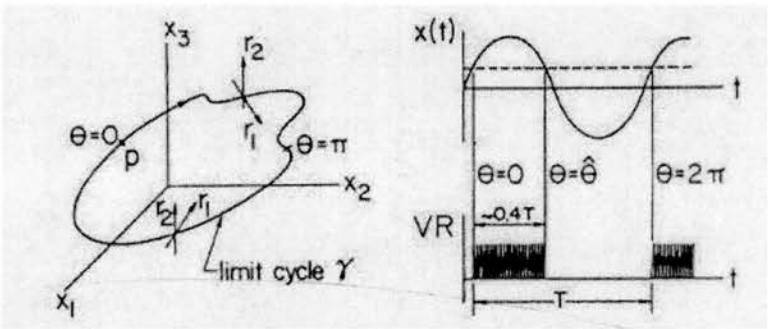


Figure 2.5: An illustration of how the limit cycle relates to the bursting activity of the ventral root. The limit cycle in a 3-D phase space (*left*). $\mathbf{r} = (r_1, r_2)$ is the amplitude deviation vector in the phase space. $\mathbf{r}$ = 0 corresponds to the points on the limit cycle. The oscillator output **x(t)** and its relation to the ventral root output VR (*right*). The dashed line represents the threshold. (Reproduced from Cohen et al. [1982] with permission.)

2. Since the individual segments burst at different frequencies when cut apart, each segmental oscillator (or a pair of such oscillators) is assumed to be coupled to its

immediate neighbors as well as possibly distant ones. A chain of these coupled oscillators (Figure 2.6) is used to describe the lamprey spinal cord. Mathematically speaking, this chain can be described by the following set of equations:

$$\begin{aligned} \Omega &= \omega_1 + H_A(\phi_1) && k = 1 \\ \Omega &= \omega_k + H_A(\phi_k) + H_D(-\phi_{k-1}) && 1 < k < n \\ \Omega &= \omega_n + H_D(-\phi_{n-1}) && k = n \end{aligned}$$

where Ω is the common frequency at which all the oscillators are locked, ω_k is the intrinsic frequency of oscillator k, ϕ_k is the phase difference between oscillator k + 1 and oscillator k, H_A and H_D are the ascending and descending coupling functions respectively. For further details, refer to [Cohen et al., 1982, Marder et al., 1997].

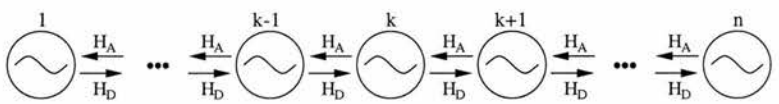


Figure 2.6: A chain of n coupled oscillators. (Reproduced from Ijspeert [1998] with permission.)

So far, most of the work deals with coupling with immediate rostral and caudal neighboring segments. One important point about the coupling is that it can speed up or slow down the receiving oscillator. Several coupling functions have been proposed to produce a frequency change as a function of the phase difference between the two coupled oscillators. The simplest one depends sinusoidally on the phase difference [Cohen et al., 1982]. Unfortunately, it was found to be inadequate for explaining the phenomenology of the lamprey spinal cord. This led to a more biologically plausible model which was constructed using details from biophysical experiments (see Williams [1992b] for example). In the improved model, the phase is again used to represent the state of an oscillator over time. It has smooth values over a period of time equal to one locomotor cycle. The coupling functions are more general than the previous one [Cohen et al., 1992].

According to Cohen et al. [1982], "For each coupling function, the phase lag at which one oscillator has no effect on the frequency of its neighbor plays a special role". This kind of phase lag is called the *zero-crossing phase lag*. It is represented by ϕ_A and ϕ_B for ascending and descending coupling respectively. An important feature of these coupling functions is that there exists a region which contains both ϕ_A and ϕ_B. In such a region, the slope of the ascending coupling function is positive while that of the descending one is negative. These restrictions are required for the stability of the solutions. In addition, if the ascending and descending coupling functions are not equal, then stable swimming with constant phase lag will result. Mathematical analysis shows that one of the coupling functions must be dominant. Further analysis showed that, except for a small boundary layer at one end, the zero-crossing phase lag of this dominant coupling is actually the constant intersegmental phase lag [Cohen et al., 1992]. Thus, the mathematical model can be used to provide a mechanism which produces constant phase lags in a way which cannot be explained by the biophysical and connectionist models.

Comparative analysis of these theoretical findings and experimental results showed contradictory findings. Evidence from forcing and split-bath experiments[7] showed that ascending coupling is dominant over descending coupling [Williams and Sigvardt, 1995]. Since it is possible to entrain from either end of the spinal cord, both types of couplings are supposed to exist. However, when forced from the rostral end (but not from the caudal end), the spinal cord failed to entrain over larger frequencies than if it were forced from the caudal end. This suggested asymmetry in coupling (favouring ascending coupling). Although most of the experimental data can be accounted for by assuming only the nearest neighbor coupling, there is evidence which suggests that long-range coupling plays a major role in intersegmental coordination [Miller and Sigvardt, 2000].

[7]Forcing experiment refers to moving the tail end of the lamprey sinusoidally to study the effect of sensory feedback. Split-bath experiment refers to perfusing the spinal cord preparation across a partitioned chamber to study the effects of local excitability on the phase coupling. Each partition can have excitatory amino acids (NMDA) of different concentrations [Matsushima and Grillner, 1992].

2.4 Ekeberg's neural model for the lamprey CPG

In this section I describe Ekeberg's two-dimensional neural model for the lamprey swimming circuitry. The model is used in conjunction with a mechanical model to allow an artificial lamprey to swim in a 2D simulation environment. The purpose of building such a neuronal-mechanical model is to study how the neural activity from the swimming CPG can be used to control mechanical movements.

Based on physiological experiments, Ekeberg [1993] hand-crafted a connectionist model for the lamprey swimming CPG. The entire network consists of 100 interconnected copies of a segmental oscillator (Figure 2.7). The segmental oscillator was proposed by Grillner et al. [1988] based on connectivity observed in the real lamprey. It consists of eight neurons of four types, namely: motoneurons (MN), excitatory interneurons (EIN), contralateral inhibitory interneurons (CIN) and lateral inhibitory interneurons (LIN). A neuron unit represents a population of functionally similar neurons in the real lamprey. Each of them receives excitations from the lamprey brainstem.

As the details of the intersegmental connections of the real lamprey CPG are not yet known, Ekeberg simplified the controller as follows. Except for the CIN neurons which have longer projections in the caudal direction, each neuron has symmetrical connections extending both rostrally and caudally. Since the neurons at both ends of the CPG receive fewer neural connections, synaptic weights are adjusted to account for this by dividing them by the number of segments a neuron receives input from.[8] The connection weights among the neurons are given in Table 2.1.

2.4.1 Mathematical modeling of neurons

In order to model a neuron unit, a leaky integrator with a saturating transfer function is used. The output u ($\in [0,1]$) is the mean firing frequency of the population the unit neuron represents. It is calculated using the following set of formulas:

$$\dot{\xi}_+ = \frac{1}{\tau_D}\left(\sum_{i\in\Psi_+} u_i w_i - \xi_+\right) \tag{2.1}$$

[8] Actually, according to Cohen (personal communication with Hallam), Ekeberg made the controller a continuous loop instead.

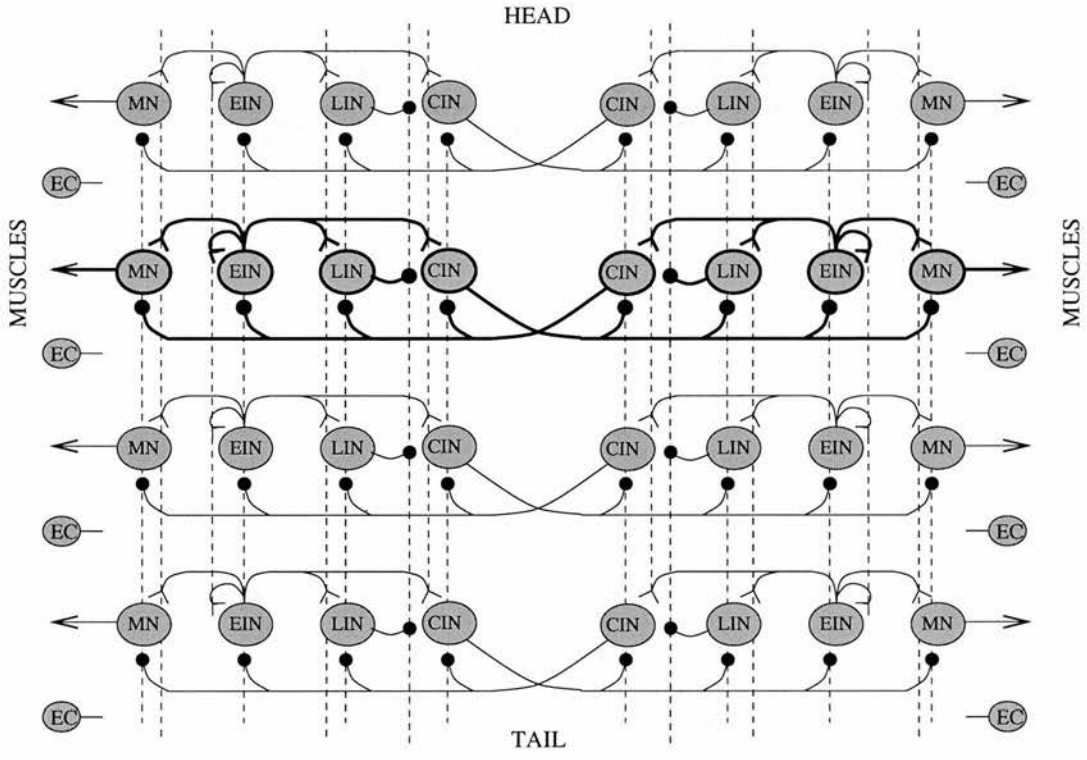


Figure 2.7: Configuration of the biological swimming controller. The controller is composed of 100 interconnected segmental oscillators (only four segments are shown here). Each segment consists of eight neurons of four types: motoneurons (MN), excitatory interneurons (EIN), lateral inhibitory interneurons (LIN) and contralateral inhibitory interneurons (CIN). Connections with a fork ending represent excitatory connections while those with a dot ending represent inhibitory connections. In addition to input signals from the brainstem (not shown here), the controller receives feedback from the stretch sensitive edge cells (EC). Note that the EC cells are not considered in my thesis. (Reproduced from Ijspeert [1998] with permission.)

From: / *To:*	EINl	CINl	LINl	EINr	CINr	LINr	BS
EINl	0.4 [2, 2]	-	-	-	−2.0 [1, 10]	-	2.0
CINl	3.0 [2, 2]	-	−1.0 [5, 5]	-	−2.0 [1, 10]	-	7.0
LINl	13.0 [5, 5]	-	-	-	−1.0 [1, 10]	-	5.0
MNl	1.0 [5, 5]	-	-	-	−2.0 [5, 5]	-	5.0
EINr	-	−2.0 [1, 10]	-	0.4 [2, 2]	-	-	2.0
CINr	-	−2.0 [1, 10]	-	3.0 [2, 2]	-	−1.0 [5, 5]	7.0
LINr	-	−1.0 [1, 10]	-	13.0 [5, 5]	-	-	5.0
MNr	-	−2.0 [5, 5]	-	1.0 [5, 5]	-	-	5.0

Table 2.1: Connection weight matrix for the biological swimming controller. Excitatory and inhibitory connections are represented by positive and negative weights respectively. Left and right neurons are indicated by *l* and *r*. *BS* stands for brainstem. The extensions from a neuron to those in neighboring segments are given in brackets. The first number indicates the number of extensions in the rostral direction while the second number indicates extensions in the caudal direction. (From Ijspeert [1998].)

$$\dot{\xi_-} = \frac{1}{\tau_D}\left(\sum_{i\in\Psi_-} u_i w_i - \xi_-\right) \tag{2.2}$$

$$\dot{\vartheta} = \frac{1}{\tau_A}(u - \vartheta) \tag{2.3}$$

$$u = \begin{cases} 1 - \exp\{(\Theta - \xi_+)\Gamma\} - \xi_- - \mu\vartheta & (u > 0) \\ 0 & (u \leq 0) \end{cases} \tag{2.4}$$

where w_i represents the synaptic weights and Ψ_+ and Ψ_- represent the groups of presynaptic excitatory and inhibitory neurons respectively. ξ_+ and ξ_- are the delayed 'reactions' to excitatory and inhibitory inputs and ϑ represents the frequency adaptation[9] observed in real neurons [Ekeberg, 1993]. The parameters for each type of neuron are given in Table 2.2. The values of these parameters and those for the connection weights are set up in such a way as that the simulation results from the model agree with physiological observations.

The neural activity of the entire swimming controller is calculated by integrating

[9]Frequency adaptation is a neural property. It means that the firing rate of a neuron is not constant for a constant input. Typically, there is a slight decrease of the firing rate over time.

Neuron type	Θ	Γ	τ_D	μ	τ_A
EIN	−0.2	1.8	30 ms	0.3	400 ms
CIN	0.5	1.0	20 ms	0.3	200 ms
LIN	8.0	0.5	50 ms	0.0	-
MN	0.1	0.3	20 ms	0.0	-

Table 2.2: Neuron parameters. Θ is the threshold, Γ the gain, τ_D the time constant of the dendritic sums, μ the coefficient of frequency adaptation and τ_A, the time constant of frequency adaptation. (From Ijspeert [1998].)

Equations 2.1 to 2.4. Ekeberg [1993] used the first order Euler method with a time step of 10 ms while Ijspeert [1998] used a fourth order Runge Kutta (RK4) method with a fixed time step of 5 ms.

2.4.2 Behavior of a segmental oscillator

The behavior of a segmental oscillator can be described as follows (refer to the highlighted segment in Figure 2.7). The brainstem provides input signals to stimulate all the neurons. Only neurons which are actively inhibited stay inactive.

Suppose that initially the neurons on the left are slightly more active. The EINl neuron excites all the ispilateral neurons while the CINl neuron inhibits all the contralateral neurons. This prevents simultaneous activity on both sides. Due to its higher threshold and time constant, the LINl neuron becomes active later in the cycle to act as a burst suppressor to the CINl neuron. This allows the neurons on the right to become active. The CINr neuron on the right in turn inhibits all the neurons on the left. After a while, the activities of the neurons on the right are terminated by the LINr neuron and the entire cycle starts again. Using this mechanism, an alternating pattern of muscular activity can be generated.

2.4.3 Behavior of the entire swimming CPG

The complete swimming CPG functions as follows: Global excitation from the brainstem stimulates all neurons in the CPG; sufficient stimulation results in oscillations in

each individual segment at a frequency that depends on the strength of this global excitation signal. Extra excitation is supplied from the brainstem to the five most rostral segments of the CPG. The effect of this, interacting with intersegmental coupling, is to induce a roughly equal relative phase lag between successive segments in the CPG, with the result that caudally traveling waves of neural activity appear. The global excitation controls the amplitude of the motoneurons as well as the frequency of oscillation of the CPG. The extra excitation alters the intersegmental phase lag largely independently from the global excitation.

2.5 Ekeberg's mechanical model on lamprey

Ekeberg [1993] proposed a 2D mechanical lamprey model to study how the muscular activity induced by the model CPG affects swimming. Since the CPG is hand-crafted using real physiological data, the model lamprey is made to approximate the size and shape of the real one. In order to describe the position and the shape of the body over time, a chain of links forming the midline is used. The model lamprey consists of 10 rigid body links with nine joints of one degree of freedom (Figure 2.8). Each link is assumed to be a cylinder with an elliptical cross-section. The link is represented by its center of mass coordinate (x_i, y_i) as well as the angle (φ_i) between it and the x-axis.

On each side of the body, muscles connect each link to its immediate neighbors. The muscles are modeled as a combination of springs and dampers [Ekeberg, 1993]. The outputs from the motoneurons control the spring constants of the corresponding muscles. As the neural wave travels along the body from head to tail, the successive contraction of muscles creates a mechanical wave. This in turn generates inertial forces from the surrounding water that propel the lamprey forward (more on this later in this chapter).

2.5.1 Body parameters

The entire length of the model lamprey is 0.3 m. Each mechanical link has an elliptical cross section of constant height (30 mm) and variable width. Its length is 30 mm. The

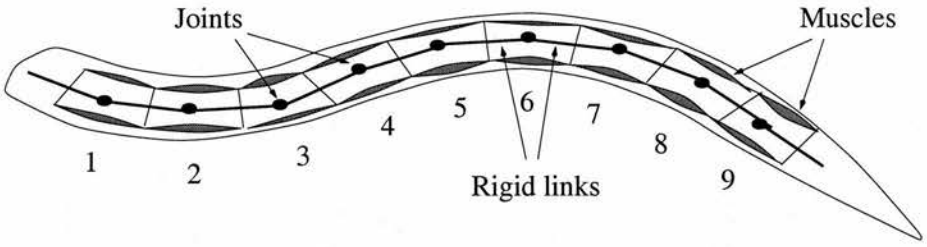


Figure 2.8: The mechanical lamprey model. (Reproduced from Ijspeert [1998] with permission.)

mass and moment of inertia of each link are calculated by assuming that the density of the lamprey is the same as that of water (Table 2.3). Note that some of the values listed in the table are later corrected by Ijspeert (refer to Subsection 2.7.1 for more information).

link	w_i [mm]	m_i [g]	I_i [g mm^2]	$\lambda_\perp$ [Ns^2/m^2]	$\lambda_\parallel$ [Ns^2/m^2]
1	20.0	4.5	45.0	0.045	0.03
2	20.0	4.5	45.0	0.045	0.02
3	20.0	4.5	45.0	0.045	0.01
4	20.0	4.5	45.0	0.045	0.0
5	17.2	3.8	35.6	0.045	0.0
6	15.0	3.15	27.5	0.045	0.0
7	11.7	2.5	20.4	0.045	0.0
8	8.3	1.8	14.2	0.045	0.0
9	5.0	1.1	8.6	0.045	0.0
10	1.7	0.45	3.4	0.045	0.0

Table 2.3: Parameters for the mechanical simulation proposed in [Ekeberg, 1993]. w_i, m_i and I_i are the width, mass and inertia of link i respectively. $\lambda_\perp$ and $\lambda_\parallel$ are the λ factors used to calculate the water forces. (Refer to Subsection 2.5.4 for details.)

2.5.2 Muscle forces

Since there are 100 neural segments in the entire swimming CPG, each mechanical link corresponds to 10 neural segments. The torque at each joint is controlled by the

averaged motoneuron outputs of 10 consecutive neural segments along the CPG from segments 6 to 95 .[10] The reason for skipping the 5 segments at each end of the CPG is that neurons at the head and the tail receive fewer inputs.

Ekeberg assumes that the length of the muscle fibers varies linearly with the local curvature of the body and that the muscular signals linearly control the forces generated by the muscles. This allows a linear relationship between the muscular activity and the spring constant. To contract the muscles, the motoneurons increase the spring constant of the muscle and reduce its resting length. The muscle torque equation is thus:

$$T_i = \alpha(M_l - M_r) + \beta(M_l + M_r + \gamma)(\varphi_{i+1} - \varphi_i) + \delta(\dot{\varphi}_{i+1} - \dot{\varphi}_i), \tag{2.5}$$

where α, β, γ, and δ determine respectively the gain (the amount of amplification of the difference in the left and right motoneuron outputs), the stiffness gain (the amount of increase in stiffness for a given average motoneuron output), the tonic stiffness (natural spring stiffness when there is no signal from the motoneurons), and the damping coefficient of the muscles. The parameters are: $\alpha = 3.0$ [N mm], $\beta = 0.3$ [N mm], $\gamma = 10.0$ and $\delta = 30.0$ [N mm ms]. Note that these values are incorrect. They are later corrected by Ijspeert (refer to Subsection 2.7.1 for more information).

2.5.3 Inner forces

In order to ensure that all the mechanical links in the model stay connected, inner forces are applied as constraints to the joints. The constraints equations are:

$$x_i + \frac{l_i}{2}\cos\varphi_i = x_{i+1} - \frac{l_i}{2}\cos\varphi_{i+1} \tag{2.6}$$

$$y_i + \frac{l_i}{2}\sin\varphi_i = y_{i+1} - \frac{l_i}{2}\sin\varphi_{i+1} \tag{2.7}$$

[10] Ekeberg [1993] and Ijspeert [1998] use the outputs of the motoneurons from a single segment of the complete CPG to drive a mechanical joint. The nine chosen segments are evenly spaced from segment 6 to 95. Although this approach allows sharper signals than averaging, the swimming should not be affected too much because the averaged amplitude only decreases slightly and the averaged frequency is about the same as the original. The burst duration in my case is longer because the averaging causes the pulse width to be broadened. For more information, refer to Subsection 5.4.4.

where $i \in \{1,...,9\}$.

These can be rewritten in a compact form as $\mathbf{g}(\mathbf{p}) = \mathbf{0}$, where $\mathbf{p}$ is a column vector including the position coordinates of all the links ($\mathbf{p} = \{x_1,...,x_9,y_1,...,y_9,\varphi_1,...,\varphi_9\}$). Based on the Jacobian of $\mathbf{g}(\mathbf{p})$, a system of linear equations can be derived for calculating the necessary forces to keep the constraint equations true (for details of the calculations, refer to [Ekeberg, 1993]).

2.5.4 Water forces

The water forces on each link depend on its speed. To simplify the mechanical simulation, the water is assumed to be stationary. It is also assumed that the parallel and perpendicular water forces can be calculated independently as

$$F_{water\parallel} = \lambda_{\parallel} v_{\parallel}^2 \tag{2.8}$$

$$F_{water\perp} = \lambda_{\perp} v_{\perp}^2 \tag{2.9}$$

where $v_{\parallel}$ and $v_{\perp}$ are the parallel and perpendicular components of speed with respect to the water. $\lambda_{\parallel} = \frac{1}{2} C_{\parallel} A \rho$ and $\lambda_{\perp} = \frac{1}{2} C_{\perp} A \rho$ are coefficients which depend on the water density ($\rho = 1000\ kg/m^3$), the area A perpendicular to movement and the drag coefficient C, which depends on the shape of the body link. $C_{\perp} = 1$ and $C_{\parallel} = 0$ for all links except those close to the head (see Table 2.3). The forces act to oppose the link's movement.

2.5.5 Motion equations for the mechanical links

Each body link is acted upon by the above three types of forces: muscular torques, inner forces and water forces. Once the forces are known, the movement of each link can be calculated by integrating the accelerations of Newton's law of motion:

$$m_i \ddot{x}_i = F_{water,i,x} + \sum F_{inner,i,x} \tag{2.10}$$

$$m_i \ddot{y}_i = F_{water,i,y} + \sum F_{inner,i,y} \tag{2.11}$$

$$I_i\ddot{\varphi}_i = \sum T_i - \sum F_{inner,i,x}\frac{l_i}{2}\sin\varphi_i + \sum F_{inner,i,y}\frac{l_i}{2}\cos\varphi_i \quad (2.12)$$

where m_i and I_i are the mass and the moment of inertia of link i, x_i and y_i are the position of the center of mass of link i, and φ_i is its orientation angle. The torques and the inner forces at the ends (i = 0 and i = 10) are all zero.

To obtain the position and orientation of the mechanical links, we need to calculate the inner force $\mathbf{f}$ which stores $(F_{inner,i,x}, F_{inner,i,y})$ first.

Let $\mathbf{v} = \dot{\mathbf{p}}$ and express the above equations in a compact form as

$$\mathbf{M}\dot{\mathbf{v}} = \mathbf{W} + \mathbf{G}\mathbf{f}, \quad (2.13)$$

where $\mathbf{M}$ is the diagonal mass matrix, $\mathbf{W}$ is the matrix for external forces (water forces and muscle torques) and $\mathbf{G}$ is the transpose of the Jacobian of $\mathbf{g}(\mathbf{p})$ (Subsection 2.5.3). Differentiate $\mathbf{g}(\mathbf{p}) = \mathbf{0}$ twice, with respect to time, giving $\mathbf{G}^t\dot{\mathbf{v}} = \mathbf{H}$.

Using this relationship and multiplying both sides of Equation 2.13 by $\mathbf{G}^t\mathbf{M}^{-1}$, $(\mathbf{G}^t\mathbf{M}^{-1}\mathbf{G})\mathbf{f} = \mathbf{H} - \mathbf{G}^t\mathbf{M}^{-1}\mathbf{W}$ is obtained. Solving this equation gives us the inner force $\mathbf{f}$. The acceleration can then be obtained through Equation 2.13. Once the acceleration is found, the velocity and position of the body links can be calculated by integration. For details of the calculations, refer to Ekeberg [1993].

2.6 AI techniques used in modeling the lamprey CPGs

Knowledge from biology and artificial life (ALife) can be pooled to further our understanding of animal locomotion such as the swimming of the lamprey. Findings from biology can provide an inspiration for the ALife researchers to design, verify and improve their animates. On the other hand, modeling and simulation of biological systems can be a tool for identifying critical experiments. In this section, some of the AI techniques used in modeling biological systems such as the lamprey are described.

2.6.1 Dynamical Neural Networks

Over half a century ago, some researchers in artificial intelligence and cognitive science moved from the more traditional symbolic to non-symbolic AI. They developed

the field of artificial neural networks (ANN) (refer to [Arbib, 1995, Anderson, 1995] for a good introduction). In this field, a network of abstract neurons are combined to solve problems in a distributed manner. Each abstract neuron receives inputs from other neurons. The weighted sum of the output from the source neurons is then applied to a transfer function (such as a sigmoid function) to produce an output of the entire network. Several learning algorithms have been developed for setting the connection weights among neurons. ANNs have been successfully applied in areas such as classification, pattern recognition, signal processing and robot control, etc. The ANN variant widely used in biologically inspired robotics is *recurrent neural networks*. It is a modification of the simpler *multilayer feedforward networks*.

Multilayer feedforward networks

In a feedforward network, the neurons are organized into layers. Inputs are presented to the neurons in the bottom layer. Computations such as the one described above are then executed inside these neurons. The results of these pre-synaptic neurons are then propagated to the post-synaptic neurons of upper layers for further computation. This process continues until the uppermost layer is reached. The neuronal outputs from this layer represent the results calculated by the entire network. Training algorithms such as *back-propagation* have been used to train these networks [Golden, 1996, Ross et al., 1998].

Recurrent neural networks

Although feedforward networks trained with back-propagation algorithms have been applied to a wide range of problems such as pattern recognition and classification, these networks (once trained) are simply static mapping from the inputs to the outputs. In order to create networks which can respond to dynamic change in input from the real world environment, researchers are building systems which can store the internal states of the networks to allow temporal information processing. The ability to respond in real time is particularly important to both biological and robotic systems. Neural networks which have feedback connections are called *recurrent neural networks* (RNNs).

Recurrent networks such as those developed by [Jordan, 1986, Elman, 1990] work differently from the feedforward networks. Rather than propagating the intermediate results in one direction, the outputs of some neurons within the network are used as feedback to the inputs of others at the next time step through the *context units* [Ross et al., 1998]. This allows the networks to exhibit temporal behaviors (time dependent outputs with discrete time steps) similar to the biological ones. The networks can have memory of what happened at previous time steps. Hence, they are capable of making responses based on the sequence of input patterns.

Although both the Jordan and Elman networks are based on feedforward networks, there are two main differences between the two. In a Jordan network, the context units receive inputs from the output layer of a previous time step. The value of each context unit is also affected by a constant which controls how much it can be affected by the previous network outputs. In a Elman network, the context units are copies of the hidden units in a previous time step, and unlike the Jordan network, the context unit is not affected by the decay term. According to Golden [1996], the advantage which the Elman network has is that if it can learn temporal sequence with a small number of hidden units, information about the entire learning history can then be compressed into the activation pattern over the hidden units.

To train recurrent networks, a modified version of Hinton's back-propagation algorithm was developed. This algorithm takes advantage of the discrete time nature of computation by unfolding the network in time [Rumelhart et al., 1986, Doya, 1995, Golden, 1996, Ross et al., 1998].

Note that the Ekeberg neural model used in this thesis is another kind of recurrent network.

2.6.2 Evolutionary Algorithms

Evolutionary algorithms are optimization algorithms inspired by natural evolution. These algorithms are stochastic and parallel in nature. The two main types are genetic algorithms [Holland, 1975] and genetic programming [Koza, 1992]. Although there are a few differences between the two, the main idea is basically the same. To solve an optimization problem, the programmer specifies a fitness function and a repre-

sentation which allows the solutions of the problems to be encoded into *chromosomes*. An initial population of randomly generated chromosomes is then generated. Next, the fitness function is applied to each chromosome, to assign a fitness value according to how close they are to an ideal target solution. The algorithm then loops through operators such as *selection, variation* (including *crossover* and *mutation*), and *removal* until either a satisfactory solution is found or a prescribed condition (such as 500 generations) has been met (refer to Subsection 2.7.2 for details). Selection of parent chromosomes involves choosing chromosomes with a probability according to the fitness value. Then, children chromosomes are created by crossover. Occasionally, mutation is applied to the child chromosomes to prevent the algorithm from converging to a sub-optimal solution. Each of the newly created chromosomes is then evaluated and given a fitness value. Finally, in order to keep the size of the population constant, the least fit individuals are removed from the population. Due to selective pressure which promotes highly fit chromosomes over less fit ones, the mean fitness of the entire population increases generation after generation.

Note that genetic algorithms are used in both Ijspeert [1998] and the work presented in this dissertation.

2.7 Ijspeert's PhD work on evolutions of lamprey locomotion controllers

Ijspeert [1998] investigated the possibilities of using evolutionary algorithms to evolve neural network controllers for a model lamprey swimming in 2D computer simulations. The motivations behind his work were:

1. To explore the search space of possible neural configurations which allow undulatory locomotion.

2. To investigate how biologically inspired controllers can be generated automatically using the GA.

Rather than hand-crafting the network connections as Ekeberg [1993] did, Ijspeert used a real number GA to search for connections and synaptic weights among the

neurons proposed by Ekeberg. A direct encoding scheme was used to allow each gene to correspond to one parameter of the neural configuration. Instead of evolving the entire swimming controller at once, he used staged evolution. In the first stage, he evolved individual segmental oscillators. In the second stage, the intersegmental couplings of 100 copies of any of the segmental oscillators of stage one were evolved. In the final stage, Ijspeert evolved sensory feedback for the complete CPG to allow the model lamprey to cross a barrier of water with local speed opposite to its swimming direction. Results show that the GA can be used to evolve swimming controllers which allow the model lamprey to swim like a real lamprey.

2.7.1 Ijspeert's corrections of Ekeberg's mechanical model

Ijspeert [1998] reported that there are some problems with the mechanical model proposed in Ekeberg [1993] (described in Section 2.5 of this thesis). In particular, given the geometry of the body, the mass of each link should be increased by a factor of π to satisfy the assumption that the density of the model lamprey is the same as water density. Furthermore, Ijspeert found that the coefficients of the water forces ($\lambda_{\parallel}$ and $\lambda_{\perp}$) should be 10 times bigger given that each lamprey segment is nearly a cylinder (i.e. drag coefficient C near one). Table 2.4 lists the corrected parameters.

In order to compensate for the increased body masses and water forces, Ijspeert increased the coefficients of the muscle torque equation (Equation 2.5) by π. The corrected equation becomes:

$$T_i = \alpha(M_l - M_r) + \beta(M_l + M_r + \gamma)(\varphi_{i+1} - \varphi_i) + \delta(\dot{\varphi}_{i+1} - \dot{\varphi}_i), \tag{2.14}$$

where $\alpha = 9.4$ [N mm], $\beta = 0.94$ [N mm], $\gamma = 10.0$ and $\delta = 94.0$ [N mm ms]. Note that I am using Ijspeert's corrected parameters in this dissertation.

2.7.2 Genetic Algorithm

Rather than using the standard binary encoding as described in Goldberg [1989], Ijspeert [1998] used the real number encoding scheme.[11] A string of real numbers between 0

[11] The GA used in this dissertation is the same as the one described here.

link	w_i [mm]	m_i [g]	I_i [g mm^2]	$\lambda_\perp$ [Ns^2/m^2]	$\lambda_\parallel$ [Ns^2/m^2]
1	20.0	14.1	1414	0.45	0.3
2	20.0	14.1	1414	0.45	0.2
3	20.0	14.1	1414	0.45	0.1
4	20.0	14.1	1414	0.45	0.0
5	17.2	12.2	1137	0.45	0.0
6	15.0	10.6	944	0.45	0.0
7	11.7	8.3	691	0.45	0.0
8	8.3	5.9	465	0.45	0.0
9	5.0	3.5	271	0.45	0.0
10	1.7	1.2	90	0.45	0.0

Table 2.4: Parameters for the mechanical simulation corrected in [Ijspeert, 1998]. w_i, m_i and I_i are the width, mass and inertia of link i respectively. $\lambda_\perp$ and $\lambda_\parallel$ are the λ factors used to calculate the water forces.

and 1 (the genes) makes up a chromosome representing a potential solution. The entire population consists of a set of individuals (the chromosomes). Initially, all of them are randomly generated. Over the course of the evolution, the GA goes through selection, variation and rejection (Figure 2.9) to produce controllers with non-decreasing fitness. These three operators are described as follows:

Selection At each generation, individuals in the entire population of size N are sorted according to their fitness. A rank-based probability is used to select a fixed number of parents. The selection is based on a probability linearly proportional to the individual's position in the sorted population. For example, the fittest individual has a probability of $P(1) = \frac{N}{1+2+...+N}$ of being chosen while the i^{th} most fitted one has a probability of $P(i) = \frac{N-(i-1)}{1+2+...+N}$ of being selected. In order to avoid self-mating, only pairs consisting of different individuals are considered. Nevertheless, the same individual can appear in several mating pairs.

Variation The crossover, mutation and pruning operators are used to create new individuals (children). The crossover operator creates pairs of children from pairs of parents. Two schemes are used here: a two-point crossover (with probability *Prob_Xover*)

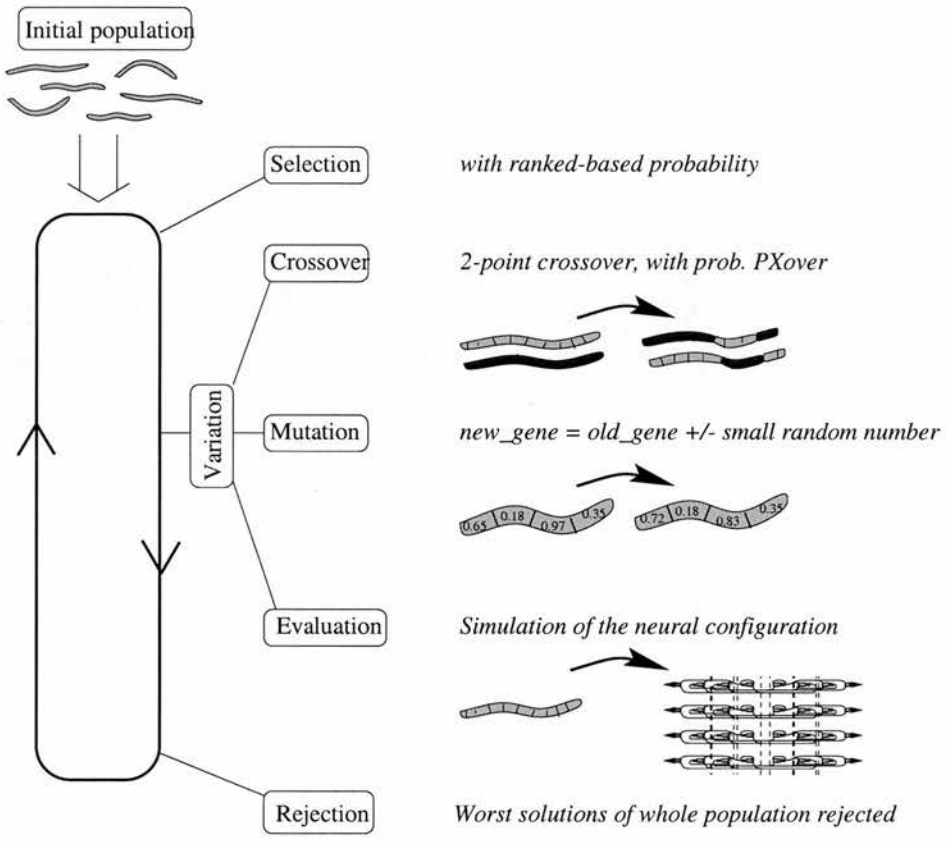


Figure 2.9: Schematic view of the genetic algorithm used throughout this thesis. (Reproduced from Ijspeert [1998] with permission.)

or simple duplication of the parents. In a two-point crossover, two points are selected randomly to divide each parent chromosome into three substrings. The substrings are then swapped to produce a new pair of children. If the selection points end up the same, a single-point crossover operation is then performed (Figure 2.10).

The mutation operator mutates each gene of the child's chromosome with a probability *Prob_mut*. It consists of adding or subtracting a random number in the mutation range as follows:

$$new\_gene = old\_gene + Mut\_Range \cdot rand$$

where *rand* is a small random number with equal distribution within [-0.5, 0.5]. If

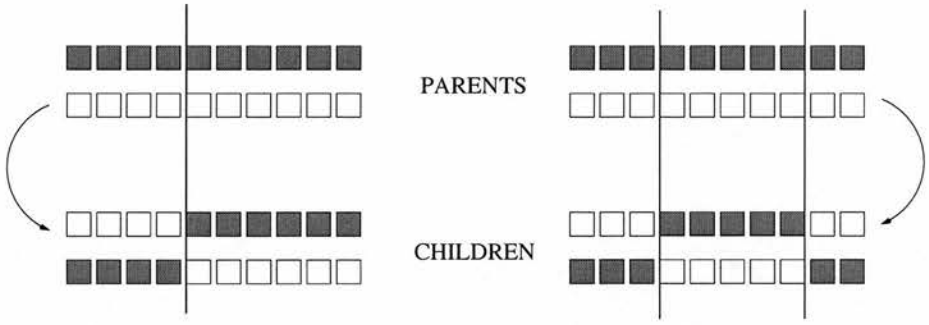


Figure 2.10: Single- and two-point crossover. (Reproduced from Ijspeert [1998] with permission.)

new_gene exceeds [0, 1], it is set to the closest boundary value. In some evolutions, a pruning operator is used to provide additional mutation. This operator is used with probability *Prob_prune* to set a gene to a value corresponding to a null connection weight. After all the newly created children are born, they are evaluated for fitness assignment.

Rejection Since the size of the population has to be kept constant, at the end of each generation, individuals with the lowest fitness from the increased population (original population plus children) are removed. In other words, the "Elitism" selection method is used.

Termination of algorithm For each set of evolutions, Ijspeert stopped all the runs at the same generation. Although it is desirable to terminate the evolutions (within the same experiment) at the same generation, this is impossible to accomplish in practice. My program is very computationally intensive and the number of jobs running on a machine has a drastic effect on the time it takes to complete one generation. Whenever it was possible, I chose to terminate the evolutions (within the same set of experiments) at the same generation.

2.7.3 Methods to evolve segmental oscillators

Ijspeert [1998] used the GA to evolve connection weights among the 8 neurons of Ekeberg's model. The same neural model is used but the sign of each neuron (excitatory or inhibitory) is not specified in advance. The fitness of each evolved oscillator is based on its ability to produce regular oscillations at a frequency depending on the level of global excitation from the brainstem.

2.7.4 GA parameters

Table 2.5 lists the parameters used in Ijspeert [1998]. Note that Ijspeert used the pruning operator to evolve segmental oscillators.

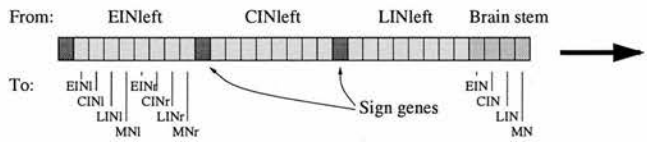
Population size	100
Number of children	30
Crossover probability	0.5
Mutation probability	0.4
Mutation range	0.2
Pruning probability	0.1

Table 2.5: GA parameters used to evolve segmental oscillators.

2.7.5 Encoding

A chromosome is used to encode the connectivity among the 8 neurons. The properties of the neurons are fixed (see Table 2.2 in Subsection 2.4.1) but the sign and connection weights are evolved. By imposing left-right symmetry, a chromosome can be represented by a string of 31 genes, each of which is a real number between 0 and 1. Each gene corresponds to one parameter of the entire neural configuration. The genes are distributed as follows (refer to Figure 2.11 for a schematic view).

Three genes are used to indicate whether the three groups of interneurons (EIN, LIN and CIN) are excitatory or inhibitory. Since the motoneurons (MN) are always excitatory, there is no need to encode their signs. Another four genes are used to



	EINl	CINl	LINl	EINr	CINr	LINr	Brain Stem
EINl	1.3	-0.8			-2.7		4.3
CINl		-3.5	-4.2				10.8
LINl	7.4						3.0
MNl	6.0				-3.0		5.8
EINr				1.3	-0.8		4.3
CINr		-2.7			-3.5	-4.2	10.8
LINr				7.4			3.0
MNr		-3.0		6.0			5.8

Figure 2.11: Encoding of a segmental network. (Reproduced from Ijspeert [1998] with permission.)

encode the synaptic weights from the brainstem to all four groups of neurons. Each gene corresponds to a value in the range between [-5, 15]. This range is chosen to include the one used in the biological model [-2, 13]. Depending on the value of the sign gene, the rest of the genes are used to encode the connection weights which correspond to real values between [-5, 0] or [0, 15]. If the sign gene is smaller than 0.5, the corresponding interneuron is inhibitory. Otherwise, it is excitatory.

2.7.6 Evaluation

Ijspeert [1998] defined a fitness function to reward segmental oscillators which exhibit the following features:

1. The production of regular and anti-phase oscillations between the left and right motoneurons with a pulse in each period.

2. Monotonic increase in amplitude and frequency with increase in global excitation from the brainstem.

3. A minimum number of synaptic connections. Connections which do not contribute to the creation of oscillatory activity are eliminated.

Since only the motoneurons have a direct effect on muscular activity, only the outputs from the MNs are used to evaluate the fitness of the segmental oscillators. This means that segmental oscillators with fewer than 8 active neurons can be evolved. During the evolution, neural simulations lasting 3000 ms are performed with asymmetric

initial conditions (i.e. all the neurons on the left are excited). Different levels of excitations from the brainstem (anchored at 1.0 and changing in steps of ± 0.1) are applied to the oscillators to determine the ranges of frequencies which the segmental networks could produce.

The fitness function which rewards these features is given as follows:

$$Fitness1 = \underbrace{fit\_var \cdot fit\_reg \cdot fit\_anti\_phase}_{fit\_oscil} \cdot fit\_freq \cdot fit\_connectivity \in [(0.05)^5, 1] \quad (2.15)$$

where

- *fit_oscil* consists of three fitness factors namely *fit_var*, *fit_reg* and *fit_anti_phase*. *fit_var* and *fit_reg* respectively reward solutions which can produce varying motoneuron output with regularity. They are defined as:

$$\begin{aligned} fit\_var &= \frac{f_1 + f_2}{2} \\ fit\_reg &= \frac{f_3 + f_4 + f_5 + f_6}{4} \end{aligned}$$

  where the f_i's are defined in Table 2.6 and Table 2.7.

  As for *fit_anti_phase*, it rewards solutions which can produce anti-phase pulses from the left and right motoneurons. Its definition is also provided in Table 2.6.

  Note that all the above factors are statistical measurements of motoneuron signals from the last second of the neural simulation. The boundaries of the transformation functions are chosen by Ijspeert [1998]. According to Ijspeert, *fit_oscil* over 0.45 clearly distinguishes oscillators which can oscillate regularly with an alternating pattern from those which cannot.

- *fit_freq* rewards solutions which produce a large range of frequencies as the excitations from the brainstem are varied. The frequency range is defined as the ratio between maximum and minimum frequencies. It is only measured if *fit_oscil* > 0.45. To obtain the frequency range, neural simulations at different levels of global excitation are conducted in steps of ± 0.1. Only ranges in which both the amplitude and the frequency of the motoneuron outputs increase monotonically with global excitation are rewarded. At each excitation level, a function similar to *fit_oscil* is used to verify that the oscillations are regular.

- *fit_connectivity* rewards solutions with fewer connections. Its value depends on the connectivity ratio (*con_ratio*) which is defined as the ratio between the number of neural connections and the maximum number of possible connections (56). Higher reward is given to those with smaller connectivity. For those with *con_ratio* less than 0.3 (i.e. fewer than 18 connections), a maximum reward of 1 is awarded.

Depending on where the value lies within the good and bad boundary, each fitness factor is assigned a value between 0.05 and 1.00 using a transformation function as follows:

$$F(x) = 0.95 \cdot \frac{x - G}{G - B} + 1.0$$

where B and G represent the bad and good boundary respectively (Figure 2.12). The variables of these fitness factors and their corresponding boundaries are given in Table 2.6.

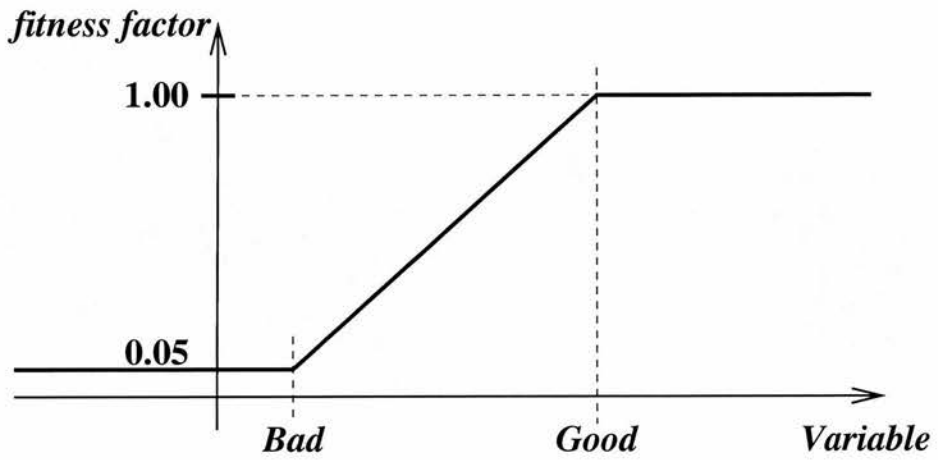


Figure 2.12: Transformation function used to calculate the value for each fitness factor. Note that the *good* boundary need not necessarily be greater than the *bad* boundary. (Reproduced from Ijspeert [1998] with permission.)

Function	Variable	[*bad,good*] Boundaries
f_1	Mean number of zeros	[3, 8]
f_2	Mean standard deviation	[0.1, 0.5]
f_3	Left-right period difference	[0.15, 0.00]
f_4	Consecutive period difference	[0.15, 0.00]
f_5	Signal difference in consecutive cycles	[0.40, 0.05]
f_6	Signal difference between left and right bursts	[0.40, 0.05]
fit_anti_phase	Left-right difference	[0.0, 0.8]
fit_freq	Oscillation frequency	[1.0, 12.0]
fit_connectivity	Connectivity ratio	[1.0, 0.3]

Table 2.6: Variables and boundaries for the fitness function. See Table 2.7 for the mathematical definition of some of these variables. (From Ijspeert [1998].)

Variable	Mathematical definition
Mean standard deviation	$\sqrt{\frac{\sum_{t=1}^{N}(U_l(t)-\bar{U}_l)^2}{2N}+\frac{\sum_{t=1}^{N}(U_r(t)-\bar{U}_r)^2}{2N}}$
Left-right period difference	$\lvert\frac{\sum_{cycle=1}^{C}P_l(cycle)}{\bar{P}}-\frac{\sum_{cycle=1}^{C}P_r(cycle)}{\bar{P}}\rvert$
Consecutive period difference	$\frac{\lvert P_l(C)-P_l(C-1)\rvert}{2\bar{P}}+\frac{\lvert P_r(C)-P_r(C-1)\rvert}{2\bar{P}}$
Signal difference in consecutive cycles	$\frac{\sum_{t\in\text{last\_cycle}}\lvert U_l(t)-U_l(t-\bar{P})\rvert}{\sum_{t\in\text{last\_cycle}}(U_l(t)+U_l(t-\bar{P}))}$
Signal difference between left and right bursts	$\frac{\sum_{t\in\text{last\_cycle}}\lvert U_l(t)-U_r(t-\bar{P}/2)\rvert}{\sum_{t\in\text{last\_cycle}}(U_l(t)+U_r(t-\bar{P}/2))}$
Left-right signal difference	$\frac{\sum_{t=1}^{N}\lvert U_l(t)-U_r(t)\rvert}{\sum_{t=1}^{N}\lvert U_l(t)+U_r(t)\rvert}$
Connectivity ratio	$\frac{\text{N\_Connections}}{56}$

Table 2.7: Mathematical definition of variables. N and C are the numbers of integration steps and simulated cycles respectively. U_l and U_r are the outputs of left and right motoneurons. $P_l(j)$ is the period of cycle j for the left motoneuron (cycles start at the onsets of the burst). $\bar{P}$ is the average period. Note that I have corrected the mistakes in the first three equations from Table 4.2 on p.81 of Ijspeert's PhD thesis. In the original table, the first two mathematical definitions did not match the descriptions given. In the first equation, Ijspeert did not put a square root in the mean standard deviation formula. In the second definition, the left-right period difference should be defined as the absolute difference (rather than the summation) of the two terms. Also, the denominators should be $\bar{P}$ rather than 2C because C represents the number of simulated cycles, which has nothing to do with the definition here. (Adapted from Ijspeert [1998] with errors in the first three equations corrected.)

2.7.7 Methods for evolving intersegmental couplings

The entire swimming CPG is created by evolving intersegmental couplings between 100 copies of any given segmental oscillator.

Genetic Algorithm

The same real number GA described in Subsection 2.7.2 is used here but without the pruning operator. The parameters for the GA are listed in Table 6.1.

Population size	40
Number of children	12
Crossover probability	0.5
Mutation probability	0.4
Mutation range	0.2

Table 2.8: GA Parameters used to evolve the intersegmental couplings. Note that these parameters will be used to evolve couplings for the efficient swimming controllers in Chapter 6.

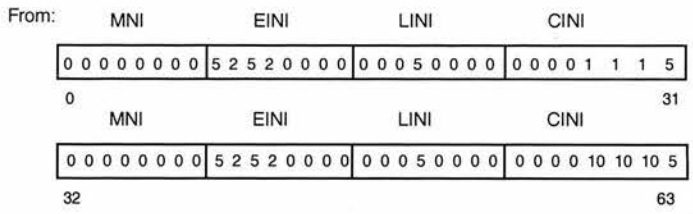
Encoding

A chromosome is used to encode the intersegmental couplings among any given segmental oscillator and its neighbors (refer to Figure 6.2). The couplings are extensions from a neuron in one segment to other post-synaptic neurons in the neighboring segments. For each neuron within a segment, the number of extensions in either the rostral or caudal direction is an integer value between 0 and 12. The reason for choosing this range is that it includes the maximum number of extensions (10) in the biological model. The number of extensions is then linearly transformed into a real number between 0 and 1 as a gene value. Due to left-right symmetry, a chromosome of length 64 can be used to encode the intersegmental connections of the entire CPG.

Evaluation

Ijspeert [1998] defined a fitness function to reward swimming controllers which exhibit the following features:

1. The production of regular and stable oscillations in all 100 segments with coordinated phase lag to allow travelling of neural waves along the body.



From / To	MNI	EINI	LINI	CINI	CINr	LINr	EINr	MNr
MNI	0	5	0	0	5	0	0	0
EINI	0	2	0	0	1	0	0	0
LINI	0	5	0	0	1	0	0	0
CINI	0	2	5	0	1	0	0	0
CINr	0	0	0	1	0	5	2	0
LINr	0	0	0	1	0	0	5	0
EINr	0	0	0	1	0	0	2	0
MNr	0	0	0	5	0	0	5	0

Rostral

From / To	MNI	EINI	LINI	CINI	CINr	LINr	EINr	MNr
MNI	0	5	0	0	5	0	0	0
EINI	0	2	0	0	10	0	0	0
LINI	0	5	0	0	10	0	0	0
CINI	0	2	5	0	10	0	0	0
CINr	0	0	0	10	0	5	2	0
LINr	0	0	0	10	0	0	5	0
EINr	0	0	0	10	0	0	2	0
MNr	0	0	0	5	0	0	5	0

Caudal

Figure 2.13: Encoding of intersegmental couplings. The couplings of each neuron to its neighbour in both the rostral and caudal directions are stored in two separate matrices. A chromosome of length 64 is used to encode the information. Elements from the first 32 genes correspond to connections stored in the matrix "Rostral". The last 32 genes correspond to connections stored in the matrix "Caudal". Note that for illustration purposes, the number of extensions represented by each gene has not been transformed to a real number between 0 and 1 in the figure.

2. Independent modulation of frequency and phase lag by varying respectively the amount of global excitation and extra excitation from the brainstem.

3. The ability to change the swimming speed by varying either the oscillation frequency or phase lag.

4. The ability to exhibit a wide range of frequencies, phase lags and swimming speeds.

In order to evaluate the performance of the controllers with the above features,

Ijspeert defined the fitness function as follows:[12]

$$Fitness2 = min\_fit\_oscil \cdot fit\_lagcontrol \cdot fit\_freqcontrol \cdot fit\_speed \tag{2.16}$$

where

- *min_fit_oscil* is the minimum of the *fit_oscil* values (see Subsection 2.7.6) at segment 1,10,20,..,100 when the controller is simulated with global excitation *excit0* and there is no extra excitation. *excit0* corresponds to the middle of the global excitation range in which the segmental oscillator oscillates regularly.

- $fit\_lagcontrol = \begin{cases} 0.05 + \frac{phase\_lag\_range1}{1+freq\_range1} & \text{if} < 1 \\ 1 & \text{otherwise} \end{cases}$

  To measure *phase_lag_range1* and *freq_range1*, neural simulations are conducted with fixed global excitation (*excit0*) and an increasing extra excitation from 0 to 300% in steps of 20% are performed. The phase lag range is non-zero only if the oscillations are regular in all segments (i.e. $min\_fit\_oscil > 0.45$) and if the phase lag increases monotonically with an increase in extra excitation. In order to reward solutions whose frequency and phase lag can be changed independently, the phase lag range is divided by the frequency range.

- $fit\_freqcontrol = \begin{cases} 0.05 + \frac{freq\_range2}{1+phase\_lag\_range2} & \text{if} < 1 \\ 1 & \text{otherwise} \end{cases}$

  To measure *freq_range2* and *phase_lag_range2*, neural simulations are conducted with global excitation varying around *excit0* and extra excitation fixed at *extra0*.[13] *Extra0* is found in *fit_lagcontrol* when *phase_lag_range1* includes a phase

[12] As these features are important for any swimming controller, my fitness functions for evolving efficient swimming controllers in Chapter 6 are based on this fitness function.

[13] I used *excit0* as the anchor and then used binary search to find the range in which the frequency of oscillations increases monotonically with extra excitation.

lag of 1%. Then *extra0* is defined as the extra excitation which corresponds to the phase lag closest to 1% (i.e. within 0.95 to 1.05%). If a phase lag of 1% cannot be found, *extra0* is undefined. Note that one reason for placing special emphasis on the 1% phase lag is that when the body forms an (approximate) S-shape, the lateral components of the thrust largely cancel each other out, leaving only the forward components of reactive force on the body. This corresponds to efficient swimming [Breder, 1926, Lighthill, 1960, Williams, 1986]. Another reason is the reduction in the search space of possible solutions.

The frequency range is non-zero only if *extra0* exists and if the frequency of oscillations increases monotonically with global excitation.

- $fit\_speed = \begin{cases} 0.05 + speed\_range & \text{if} < 1 \\ 1 & \text{otherwise} \end{cases}$

  speed_range is the range of swimming speeds in [m/s] achieved with normal body scale during the simulations performed in *fit_freqcontrol*.

The ranges of frequency, phase lag and speed mentioned above are the absolute difference between the maximum and minimum non-negative values after normalization by the corresponding fixed target value: 2.5% for the phase lag range, 10.0 Hz for the frequency range and 0.8 m/s for the speed range.

2.8 Summary

The study of animal locomotion is of interest to both biologists and robotics researchers. It is believed that the CPGs are responsible for generating constant locomotion gaits while inputs from the brain and sensory feedback provide modulations. As animal locomotion requires coordination of many actuators, the question of how input commands from the brain can change the direction and speed of movement is interesting. From studies of vertebrates and invertebrates, it appears that motor programming, command systems, afferent regulation and neuromodulation are the four common principles underlying motor control. The lamprey is chosen as an experimental subject in research on vertebrate locomotion due to its simplicity and similarity to higher vertebrates. Over the past 15 years, several models at different levels of complexity have been proposed for the motor control of the lamprey. Each of them has advantages and disadvantages.

In this chapter, I presented Ekeberg's 1993 neural and mechanical model for the lamprey. The neural model is based on neurophysiological data. The segmental network is similar to that proposed by Grillner et al. [1988], while the intersegmental coupling is simplified so that each segment has symmetrical connections to several neighboring segments in both rostral and caudal directions. One exception is that the connections from the CIN neurons have asymmetrical couplings which favor the caudal direction. As regards the mechanical model, some of the parameters are calculated incorrectly. They are later corrected by Ijspeert.

Finally, as this thesis is inspired by Ijspeert's PhD dissertation on evolving swimming controllers for the lamprey, some of the related AI techniques (such as neural networks and genetic algorithms) as well as descriptions of his work are introduced.

It is believed that the interaction between the biologists, computational neuroscientists and ALife researchers can lead to a better understanding of animal and animate locomotion control.

Chapter 3

Reimplementation and quantitative analysis of Ekeberg's neural and Ijspeert's mechanical model

In this chapter, I describe my reimplementation of Ekeberg's neural and Ijspeert's mechanical model. The post-synaptic normalization scheme used to implement the swimming CPG is described, followed by the quantitative analysis of both the segmental oscillator and the entire swimming controller. As the lamprey can swim in any direction, a new speed calculation algorithm which takes turning into account is presented. This is followed by a description of the procedure for neuromechanical simulation which I use to generate various surface plots for visualizing how the characteristics of the neural wave and swimming performance (such as speed and efficiency) vary under different excitation combinations from the brainstem. Given that I am trying to use the results from neuromechanical simulations as inspiration for real biological experiments, a discussion of the realism of my neural and mechanical simulations concludes this chapter.

3.1 My implementation of Ekeberg's neural model

I wrote the neural simulator in C programming language to implement Ekeberg's neural model described in Section 2.4. To calculate the neural activity of either the segmental oscillator or the entire swimming CPG, I used the fourth order Runge-Kutta (RK4) method to integrate the system of coupled differential equations (Equations 2.1 to 2.4). The reason for choosing the RK4 method rather than the Euler method as mentioned in Ekeberg [1993] is that the former produces more accurate results. Instead of using an integration time step of 5 ms as Ijspeert [1998] did, I chose a time step of 1 ms to better match the mechanical simulator (which runs at a time step of 1 ms), and for higher accuracy.

Based on experience, it takes approximately two simulated seconds for the neural simulator to stabilize. Given that the mechanical simulator requires a similar amount of time to stabilize and that I want to measure the swimming speed when the fish is cruising (swimming at a relatively constant rate), each neural simulation lasts for 8 seconds.

3.1.1 Implementation of the CPG using post-synaptic normalization scheme

To implement the intersegmental neural connections within the CPG, I used a post-synaptic normalization scheme. The reason for normalizing the connections is that neurons at the ends of the CPG receive fewer connections (inputs) compared with those located in the middle of the network. As a result, the connection weights need to be calibrated. For illustration purposes, consider the connections from the EINl neurons to other EINl neurons in the first segment. Under this scheme, one takes the number of inputs to a unit, then the total input is divided by the number of units contributing to it, and this becomes the input to use. In other words, each unit receives the average input of all those that connect to it (Figure 3.1).

Since the EINl neurons can connect to the EINl neurons of two neighbouring segments in the rostral direction (Table 2.1), there is a connection from the EINl neurons at segment 3 to those at segment 1. Similarily, the EINl neurons in segment 2 can

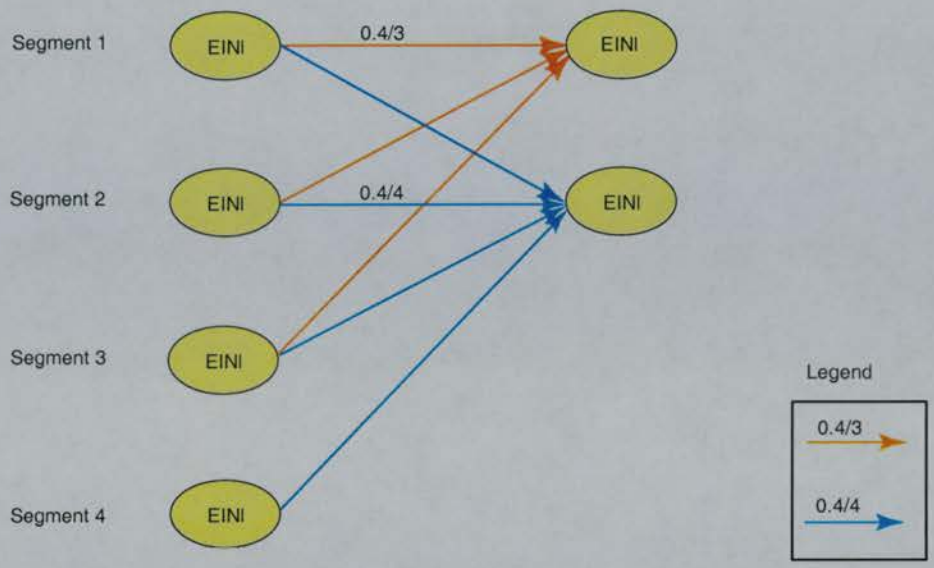


Figure 3.1: Illustration of post-synaptic implementation. To avoid confusion, only connections described in the text are shown.

connect to the EIN neurons at segment 1. Within the same segment, the EINI neurons connect to themselves. Hence, the total number of inputs to the target EINI neurons at segment 1 is three. The calibrated connection strength is therefore 0.4/3.

Using the same reasoning, I have computed the calibrated weight matrix for the biological controller. The results are shown in Table 3.1.

3.2 Quantitative analysis of a segmental oscillator

A quantitative analysis of my implementation of Ekeberg's segmental oscillator model follows.

In order to determine how the segmental oscillator behaves under different excitations, I tested my implementation of Ekeberg's segmental oscillator with a brainstem excitation from 0.2 to 1.0 in steps of 0.1. For each excitation value, a neural simulation is performed. At the end of each simulation, the amplitude and frequency from the outputs of the left motoneurons are calculated. The outputs from each neuron over

From	To	Post-synaptic normalization scheme
EINl	MNl	1.2 [1.0]
CINr		-2.0 [-2.0]
EINl	EINl	0.4 [0.4]
CINr		-2.0 [-2.0]
EINl	LINl	13.0 [13.0]
CINr		-1.0 [-1.0]
EINl	CINl	3.0 [3.0]
LINl		-1.0 [-1.0]
CINr		-2.0 [-2.0]
BS	MNl	5 [5]
	EINl	2 [2]
	LINl	5 [5]
	CINl	7 [7]

Table 3.1: Calibrated weight matrix for the Ekeberg biological controller under the post-synaptic normalization scheme. Numbers in brackets represent connections in the middle of the CPG; the others represent connections at the head. Excitatory and inhibitory connections are represented by positive and negative weights respectively. Left and right neurons are indicated by *l* and *r*. *BS* stands for brainstem. Due to symmetry between left and right, the connection weights of neurons from the right are not shown. Note that for simplificity, I assume that the input to each source neuron is 1.

time can optionally be stored in a data file for visualization under MATLAB.

When the segmental oscillator with asymmetric initial conditions[1] receives enough excitation from the brainstem, an alternating pattern of neural activity is generated (Figure 3.2).

Increasing the brainstem excitation from 0.2 to 1.0 increases the maximum amplitude as well as the frequency of oscillation of the motoneuron outputs. The amplitude ranges from 0.38 to 0.81 while the oscillation frequency ranges from 1.64 to 5.49 Hz (Figure 3.3).

A comparison of Figure 3.3 in this thesis with Figure 3.4 in Ijspeert [1998] reveals

[1] $\xi_+(0) = 1$ and $\xi_-(0) = 0$ for all the left neurons and $\xi_+(0) = \xi_i(0) = 0$ for all the right neurons.

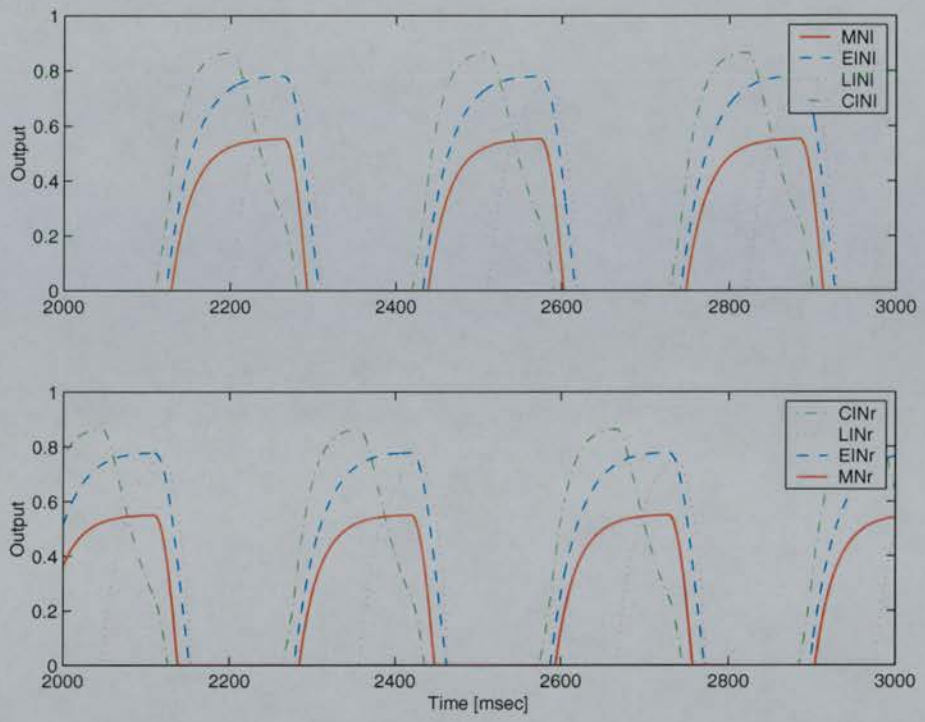


Figure 3.2: Output of neurons from a segmental oscillator with an excitation of 0.4.

that both Ijspeert and I obtained the same results. This implies that for the neural simulation of the segmental oscillator, an integration time step of 5 ms is small enough to produce accurate results.[2]

3.3 Quantitative analysis of the entire swimming CPG

To determine how the entire swimming CPG behaves under different excitation combinations, I tested my implementation of Ekeberg's model under global excitation ranges from 0.1 to 1.0 (in steps of 0.1) and extra excitation ranges from 0% to 200% (in steps of 10%). It should be noted that global excitation is the excitation which the brainstem applies to all the neurons in the CPG. Extra excitation is the excitation applied only to the neurons located in the five segments closest to the head. The extra excitation is a

[2]Unfortunately, Ekeberg has never provided a similar analysis to allow a comparison of the three implementations.

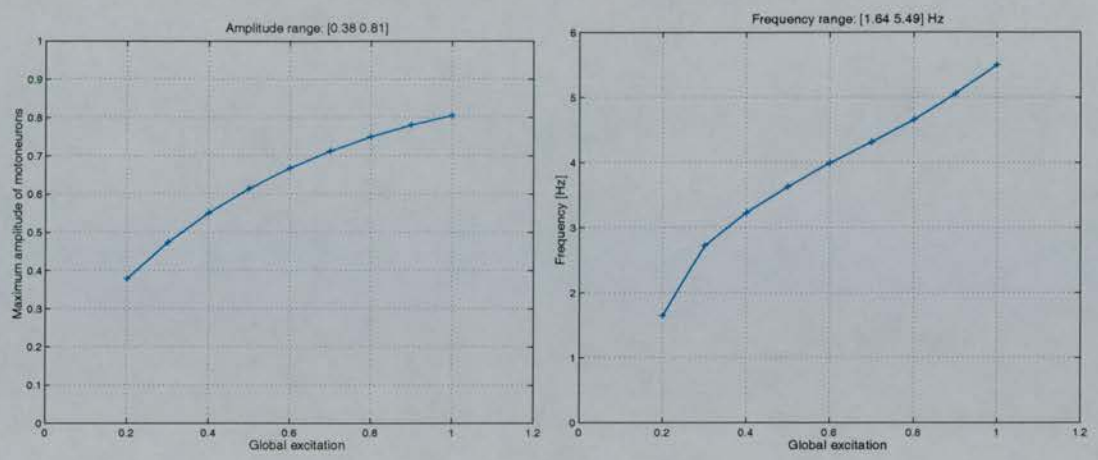


Figure 3.3: Effect of excitation from the brainstem on the amplitude of motoneuron output (*left*) and the frequency of oscillation (*right*) of the biological segmental oscillator.

percentage of the global excitation. At each excitation combination, a neural simulation is performed. The characteristics of the resulting neural wave (such as amplitude, frequency and phase lag) from the outputs of the left motoneuron in the middle of the CPG, say segment 50, are recorded. The results are shown in Figure 3.4.

The amplitude surface shows that as the global excitation increases, the amplitude of the motoneuron outputs increase. Amplitude stays fairly constant when the extra excitation is increased. Note that when the excitation is too low (at 0.1 for example), the network sometimes cannot produce regular pulses. Observation of the motoneuron outputs from the CPG under excitation combination (0.1, 110%) shows that this is the cause of the dip in the figure.

Similar to what we observe from the amplitude surface, the frequency surface shows that as the global excitation increases, the frequency of oscillation increases. Although there is a slight increase in frequency with extra excitation, frequency stays fairly constant. On the other hand, the phase lag surface indicates that the phase lag increases with extra excitation but stays fairly constant with increase in global excitation. Note that the empty regions in all three surfaces correspond to quantities that cannot be measured. In this thesis, I consider the outputs from the motoneurons to be valid only when both the left and right signals oscillate and return to zero. The reason for this is that the time instances when pulse onsets appear are required in order to calculate the

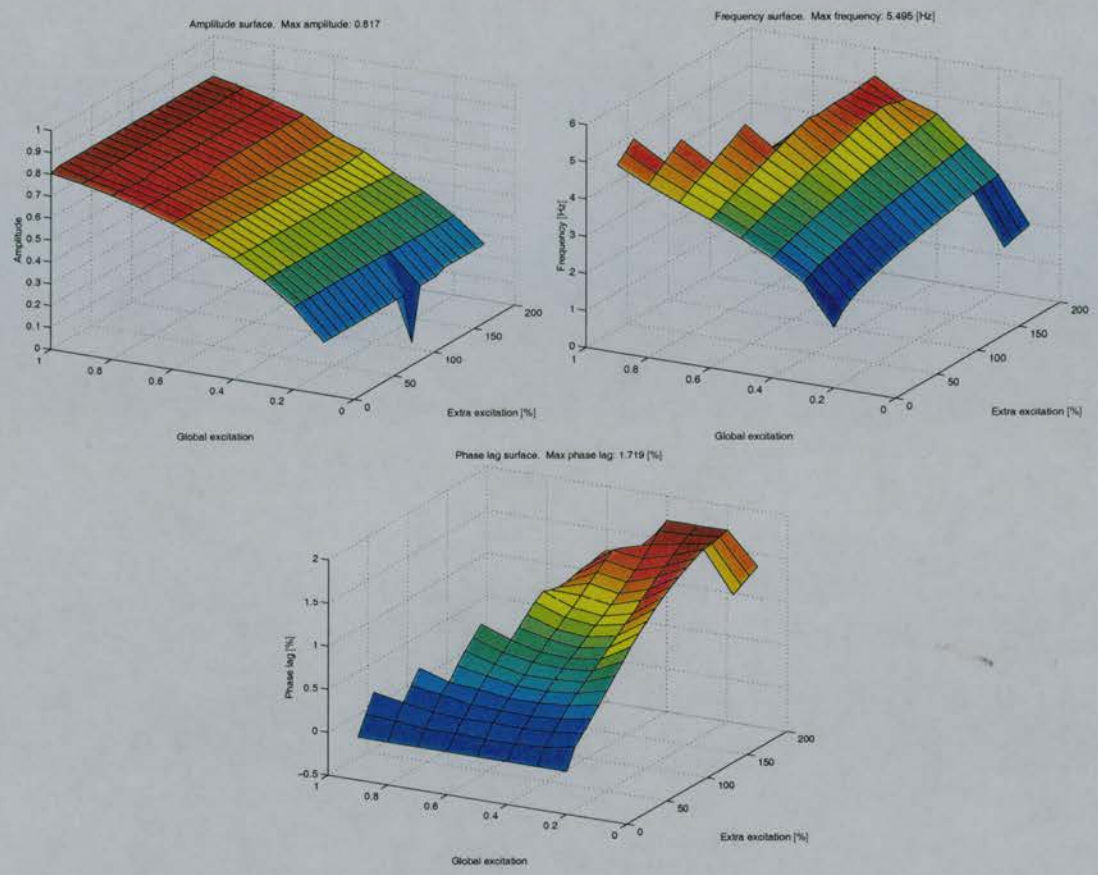


Figure 3.4: Effect of the global and extra excitations on the amplitude of motoneuron output (*top left*), frequency of oscillation (*top right*) and the phase lag (*bottom*) of the swimming CPG.

frequency and phase lag.[3]

The frequency and phase lag surfaces show that the frequency of oscillation and the wavelength of undulation can be changed almost independently. Hence, similar to the real lamprey, the neural model is capable of producing a range of oscillation

[3]Note that although there are empty regions in the frequency and phase lag surfaces at the ends of the global excitation ranges, there are none in the amplitude surface. This is because the routine CalAmplitude serves two functions in my program, depending on how it is called. It can be used to find the maximum outputs from the motoneurons regardless of whether the pulses return to zero or not, or the maximum outputs only when the pulses are oscillating and return to zero. It was discovered later that an error was made in calling this routine, so that instead of returning the maximum amplitude only when two pulse onsets were found, the routine always returned the maximum. The empty regions of the amplitude surface should correspond to those in the frequency surface. Note also that when calculating phase lag, neural waves from two segments are required. Thus, if any one of the waves has signals oscillating but never returning to zero, the phase lag is undefined.

frequencies while keeping the wavelength of undulation constant.

A comparison with Ijspeert's implementation of the swimming CPG shows that quantitatively speaking, our results are similar.[4] The qualitative differences may be due to the different ways of connecting the motoneuron outputs to the muscles. (Refer to Subsection 2.5.2 for details.)

3.4 My implementation of Ijspeert's mechanical lamprey model

The model lamprey is based on the one described in Ijspeert [1998] (originally proposed in Ekeberg [1993] with body parameters corrected. For more details, refer to Section 2.5 and Subsection 2.7.1 respectively). The fourth-order RK method with a fixed integration time step of 1 ms is used to solve the motion equations for the lamprey (see Subsection 2.5.5).

As mentioned in Section 3.1, it takes about two simulated seconds for both the neural and mechanical simulations to stabilize. As a result, the outputs of the motoneurons from only the last six seconds of the neural simulation are used to control the model lamprey. Before the simulation, I have the option of recording the coordinates and orientation of each body link into a data file. MATLAB programs are used to animate the swimming.[5]

Note that there is an implementation difference in regard to how the motoneuron outputs control the mechanical joints (refer to Subsection 2.5.2 for more information).

[4] A comparison of Figure 3.4 in this thesis and Figure 3.5 in Ijspeert [1998] shows that Ijspeert's implementation of Ekeberg's model is able to achieve a larger phase lag than my implementation (2.4% and 1.7% respectively). However, the maximum oscillation frequencies are almost the same (5.6 and 5.5 Hz respectively). Ijspeert does not provide an amplitude surface for comparison.

[5] Except for the lamprey drawing routine written by Ijspeert, all the programs are written by me.

3.5 A new speed calculation algorithm

In this dissertation, two methods are used to calculate swimming speed.[6] In the first method, the coordinates of the lamprey head during the last 2000 ms are used to calculate the direct distance traveled by the lamprey within a given period. Since the head oscillates during swimming, the swimming speed is computed from the difference in head positions taken at two instances when the lamprey body are "in-phase".[7] This method assumes that the lamprey is swimming along a straight line. It does not matter whether the lamprey is swimming straight along the x-axis or at an angle.[8] Although the fitness function plus the imposed left-right symmetry used to evolve the controllers favors straight forward swimming, the lamprey can still turn in any direction (see Figure 5.12 for an example). This should normally happen only if the oscillations are irregular or if the phase lags are too small for the 9-joint mechanical body to bend in such a way that the lateral forces along the body cancel out. In order to take turning into account, the following new speed calculation algorithm is used:

1. Scan the last of the 8 seconds of neural wave (averaged motoneuron outputs which control the first body joint) for t_1 and t_2 (where t_1 and t_2 are defined as the time instances at which the lamprey body are "in-phase"). These two time instances also correspond to the first and last pulse onset of the pulse train (within the last second of the simulation) produced by the left motoneurons of segment 1. If the two pulse onsets cannot be found, then the swimming speed is set to zero.

2. As the head is the least oscillating body point, it can be used to define the swim direction. Find the normalized forward vectors $\vec{f}_1(t_1)$ and $\vec{f}_2(t_2)$.

   - To find $\vec{f}_1$:

     a) collect four pairs of data points by recording the head locations every

[6] Except for the original experiments reported in Or and Hallam [2000] and Or et al. [2002] the rest of the experiments use the new speed calculation algorithm.

[7] By "in-phase" it is meant that the same lamprey body at two time instances forms the same shape. Note that Ijspeert [1998] did not take this into consideration when computing the swimming speed.

[8] Since both the neural and mechanical simulators require time to stabilize, the lamprey can end up swimming straight at an angle even when it is supposed to swim straight along the x-axis.

100 ms during the [t_1 - 300, t_1] period.

b) apply the least square method to the data points to interpolate a regression line l_1.

c) obtain the projection of the head position at t_1 - 300 ms on l_1.

d) obtain the projection of the head position at t_1 on l_1.

e) using the projections, construct a normalized forward vector $\vec{f}_1$ that points along l_1. This vector points to the swim direction.

- To find $\vec{f}_2$:

  a) record the head locations every 100 ms during the [t_2 - 300, t_2] period.

  b) apply the least square method to the head locations to find the regression line l_2.

  c) obtain the projection of the head position at t_2 - 300 ms on l_2.

  d) obtain the projection of the head position at t_2 on l_2.

  e) using the projections, construct a normalized forward vector $\vec{f}_2$ that points along l_2.

3. Construct a normalized direction vector $\vec{f}_3$ which indicates the orientation of the lamprey:

   a) apply the least square method to the center of mass (CM) of $link_2$ to $link_5$ at t_1 to obtain an interpolated line l_3.

   b) obtain the projection of CM($link_2$) on the interpolated line l_3.

   c) obtain the projection of CM($link_5$) on the interpolated line l_3.

   d) using the projections, construct a normalized vector $\vec{f}_3$ that points along the interpolated line from the $link_5$ projection towards the $link_2$ projection. The head of this vector indicates the heading direction of the lamprey.

4. Find the displacement L using the head positions at t_1 and t_2.

5. Calculate the turning angle $\Delta\theta$ using the normalized forward vectors $\vec{f}_1$ and $\vec{f}_2$.

$$\Delta\theta = acos(\vec{f}_1 \cdot \vec{f}_2) \tag{3.1}$$

6. Calculate the arc length C [mm] (Figure 3.5). If the turning angle is close to zero, then the arc C is a straight line. C is therefore the same as the displacement L. In other words, C is the distance between the head positions at t_1 and t_2.

   Otherwise,

$$C = \frac{L \cdot \Delta\theta}{2sin(\frac{\Delta\theta}{2})}. \tag{3.2}$$

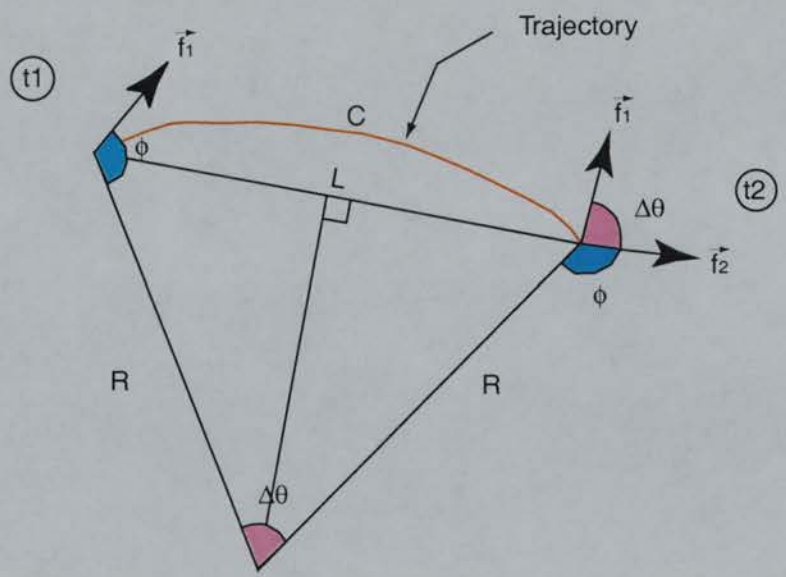


Figure 3.5: Schematic diagram of speed calculation

7. Find the dot product between $\vec{f}_3$ and the displacement vector. If the dot product between these vectors is negative, then the lamprey has turned more than 90^o in the < 1 second window. In that case, the sign of the swimming speed is negative. Otherwise, both vectors point to the same quadrant.

8. Calculate the forward swimming speed [m/s] using

$$\frac{C}{(t_2 - t_1)} \tag{3.3}$$

Depending on the application, the unit is sometimes converted to body length per second [BL/s]. If required, the angular speed [degrees/s] can be calculated by using:

$$\frac{\Delta\theta}{(t_2 - t_1)} \cdot (\frac{180}{\pi}) \cdot 1000 \tag{3.4}$$

Note that the reason for multiplying the equation by 1000 is that t_1 and t_2 are expressed in [msec].

3.5.1 Assumptions for the speed calculation algorithm

The algorithm proposed in this section assumes the following:

1. The normalized forward vectors $\vec{f}_1$ and $\vec{f}_2$ are in-phase. Otherwise, they produce incorrect speed results.

2. The lamprey moves in a circular arc on average. Note that a straight line is an arc with no turning angle.

3.5.2 Advantages of the new speed calculation algorithm

Under the new speed calculation algorithm, the swimming speed can be calculated correctly regardless of swimming direction and turning. The algorithm works for both forward and backward swimming. Since the algorithm takes turning into consideration, the resulting speed can be higher than that calculated using the old method which only considers the direct distance between the head position at two time instances and so underestimates the speed.

3.5.3 Reasons for choosing the maximum speed as a basis of comparison

In this dissertation I often compare the performance of the swimming controllers in terms of maximum achievable speed. There are two main reasons:

- The fitness functions used in Ijspeert [1998] and this thesis favor controllers which allow the lamprey to exhibit a wide range of swimming speeds. For instance, the fitness factor *speed_range* of Equation 2.16 (of Subsection 2.7.7) only considers the positive swimming speeds. Therefore, a comparison of the controller's performance based on the maximum speed means comparing the range of speeds at which they can perform.

- In Or and Hallam [2000], I investigated the robustness of a few controllers based on swimming speed against variations in body scales. Two of the controllers (controllers 2 and 3) were evolved in Ijspeert [1998] using Equation 2.16.

3.6 Neuromechanical simulations

I have written a program which accepts any controller as the input and then performs a set of neuromechanical simulations under any given range of global and extra excitation combinations. The results of the simulations can be saved for surface plotting for further analysis. Details of the procedure are listed as follows:

- *Stage 1: Uploading the controller under investigation.*
  As mentioned in Subsections 2.7.5 and 2.7.7, the segmental and intersegmental connections of each controller are encoded as two separate chromosomes. During the evolutions, these chromosomes are saved into data files. My program allows the user to specify the filenames of these chromosomes as well as the ranges of global and extra excitations as command line inputs. Once the chromosomes are loaded, they are decoded into three different connection matrices, one for segmental connections and the other two for intersegmental connections (there are two matrices for intersegmental connections because rostral and caudal connections are considered to be different and are therefore stored separately).

- *Stage 2: Neuromechanical simulations.*

  A set of neuromechanical simulations at different global and extra excitation combinations are performed. By default, global excitation ranges from 0.1 to 1.0 in steps of 0.1, while extra excitation ranges from 0 to 200% in steps of 10%. Each neuromechanical simulation requires eight seconds of neural simulation followed by six seconds of mechanical simulation.

  The motoneuron outputs from the last six seconds of the neural simulations are stored in a matrix (which can be saved in a file for visualization purpose) to drive the model lamprey in the mechanical simulation. The reason for skipping the first two seconds of neural waves is that it takes about that much time for the neural simulation to stabilize. The situation for the mechanical simulation is similar. Taking data from the last second of the remaining four seconds of the combined simulation allows me to take the measurements when the simulations are stable.

  When calculating the characteristics (such as amplitude, frequency and phase lag) of the neural wave at the end of each neural simulation, the outputs from the left motoneurons of segment 50 during the last second of the neural simulations are used. When computing the swimming speed (see Section 3.5), the head positions during the last second of the mechanical simulation are used.

- *Stage 3: Collection of data for analysis.*

  At the end of each neuromechanical simulation, the neural characteristics and swimming speed are saved into a data file for further analysis.

- *Stage 4: Visualization of results.*

  In order to visualize how the neural characteristics and swimming speed vary at different excitation combinations, I plot the amplitude, frequency, phase lag and speed surfaces. A computer animation program has also been written to display the lamprey locomotion at any specific set of excitation combinations.

When I set the global excitation and an extra excitation to 0.6 and 100% respectively,[9] a wave of neural activity traveling caudally along the spinal cord is produced. The neural wave has a maximum amplitude of 0.645, frequency of 5.03 Hz and phase lag of 1.05%. Since the phase lag is just over 1%, the next wave appears in segment one before the complete wave of undulation reaches the tail from the head (see Figure 3.6).

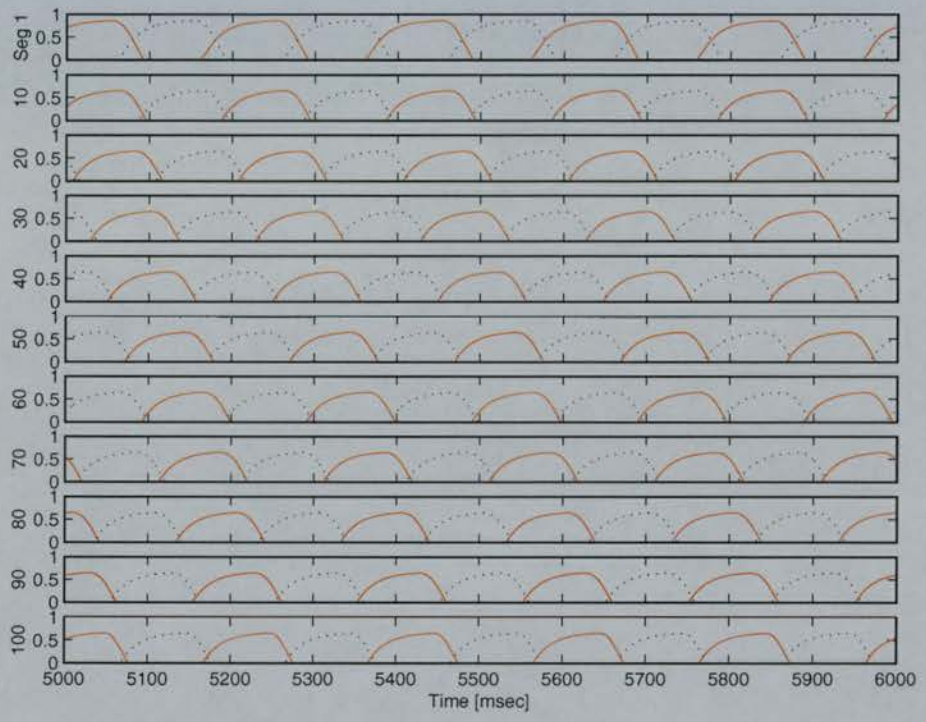


Figure 3.6: Output of the left and right motoneurons along the swimming CPG. The solid lines represent the outputs from the left motoneurons while the dashed lines represent the outputs from the right motoneurons. The global excitation is 0.6 and extra excitation is 100%. The segments oscillate at 5.03 Hz with a phase lag of 1.05%.

Applying this neural wave to the model lamprey leads to anguilliform swimming like that observed in the real lamprey (Figure 3.7).

The surface for swimming speed at different global and extra excitations is plotted (Figure 3.8). The figure indicates that when there is no extra excitation, the fish

[9]Each of the first 5 rostral segments receives an excitation of 1.2 while the remaining segments each receives an excitation of 0.6.

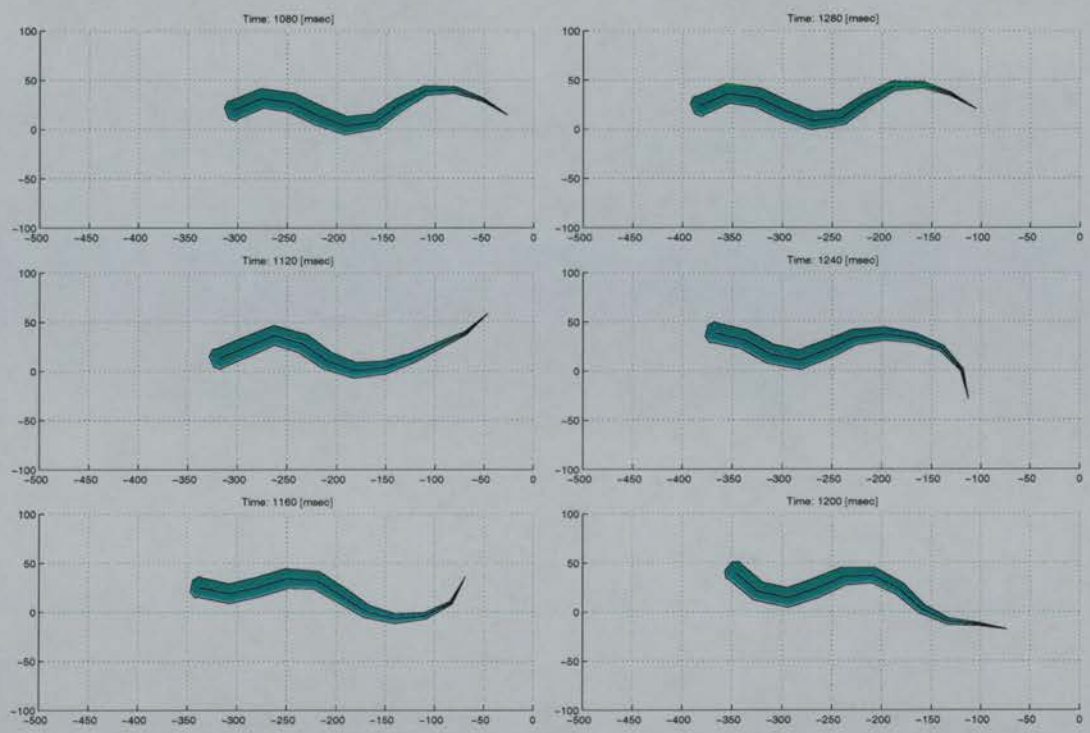


Figure 3.7: Simulated swimming of the lamprey. The swimming cycle is 200 msec. There is a 40 msec time step between each snapshot. The length and width of the swim-mill are both in [mm]. From the figures (counter-clockwise from the top-left figure), a mechanical crest is traveling from the head to the tail in one swimming cycle.

swims backward. This is due to the wriggling effect.[10] However, when there is enough extra excitation to cause a lag between the segments, increasing the global excitation increases the forward speed. Note that the speed stays fairly constant even with further increase in extra excitation. Again, the empty regions correspond to speeds that cannot be measured.[11]

A comparison of Figure 3.8 with that obtained by Ijspeert (Figure 3.9 in Ijspeert [1998]) shows that our lampreys perform similarly. However, the maximum speed achieved by his model lamprey is higher than mine (0.5 vs 0.45 m/s). The differences

[10]When there is no extra excitation, the lamprey behaves just like a fish flexing its body on the floor. The lamprey can therefore move in any direction.

[11]For example, when the two pulse onsets (which are required to determine the time instance at which the same lamprey body is in-phase) cannot be found within a second, the speed is set to zero. The plotting routine skips any excitation combination which corresponds to such invalid speed.

Figure 3.8: The effect of the global and extra excitations on the swimming speed

are due to the ways we connect the motoneuron outputs to the muscles (see Subsection 2.5.2), the different methods used to calculate the swimming speed (Section 3.5) as well as the resolutions of the excitations under investigation.

3.7 Discussion of the realism of the neural simulation

Although the Ekeberg biological controller is able to behave like the real one (for instance, oscillation frequency increases with global excitation while independently the phase lag increases with extra excitation), one should realize that it is a simplified model. As it is hand-crafted based on real physiological data, it is biologically plausible to some extend. However, some of the differences between the real system and the artificial system are as follows:

- Although some investigations have found that the connection strength between neurons decreases as the distance increases [Williams, 1992a], the distribution

of the synaptic strengths is not yet fully understood. Wadden et al. [1997] incorporated this idea of *synaptic spread* into their unsegmented biophysical network model. In this model, each synaptic strength is calculated as a function which exponentially decreases as a function of the distance between the neurons. Although their model was able to reproduce most aspects of intersegmental coordination as observed in the real lamprey, the phase lag increased with oscillation frequency. This relationship was maintained even though they used different synaptic strengths and coupling functions, but the cause of this relationship is not clear. Future work could incorporate synaptic spread into the Ekeberg connectionist model to investigate whether the independence between phase lag and frequency (as achieved by the current model) can still be maintained.

- The NMDA pacemaker properties are not included in Ekeberg's model. As a result, the network is unable to produce very slow rhythms [Ekeberg, 1993].

- According to Wadden et al. [1997], "*in the real lamprey motoneurons and interneurons are arranged in longitudinal cell columns without any apparent subdivision in segments (Wallén et al. 1985; Grillner et al. 1991), the exception being that the motoneurons in each segment send their axons into a common ventral root, and conversely afferents enter the spinal cord through a common dorsal root. In keeping with this there are no segmental subdivisions in the model, except through the arrangment of the M cells (motoneuron equivalents).*" However, for simplicity and to aid comparison of experimental results, I divided the CPG into segments as Ekeberg and Ijspeert did.

- In the real lamprey, many of the rhythmically active spinal neurons are found in different percentages at different parts of the spinal cord [Miller and Sigvardt, 2000]. For example, the LIN neurons are concentrated in the rostral segments (personal communication with Hellgren).

3.8 Discussion of the realism of the mechanical simulation

As the mechanical body is the interface between the swimming CPG and the real world, the realism of the mechanical simulation can affect the experimental results (swimming speed and efficiency for example).

Some of the differences between the reality and the artificial system are as follows:

- The mechanical model used here has only 10 segments, each of which is controlled by the averages of the motoneuron outputs from 10 neural segments (the real lamprey has about 100 body segments each of which is controlled by the left and right motoneuron outputs from the corresponding segment). In other words, the mechanical model has a finite and discrete number of joints (specifically, nine). As a result, neural waves with shorter wavelength (large phase lag) cannot propagate along the body smoothly because the model lamprey cannot bend in a continuous fashion. As the body shape cannot be as smooth as the real one in some cases, some of the results may be affected. For example, when calculating the mechanical wavelength (below and in Chapter 6), I make sure that the second to fifth body links oscillate up and down (when the fish is swimming along the x-axis) to compute the mechanical wavelength. The wavelength of a lamprey with more body segments can be different.

- In reality, time is continuous, while in simulation, time is finite (time step is 1 ms). This factor is important when sensory feedback is involved because the lamprey has to react according to stimuli from the real world. If the time step is not small enough, the lamprey will not be able to react appropriately. As there is no sensory feedback in my lamprey system, this should not affect the results much apart from possible quantisation and truncation errors.[12]

- Although it would be more realistic to simulate a 100 segment lamprey model, we have to bear in mind that as the complexity of the lamprey model increases,

[12]Quantisation error refers to the error caused by dividing time into discrete intervals. The truncation error is error due to the integration method. Ideally, we would prefer to use a higher order RK method.

simulation time increases drastically. In the program, there are two calls of Gaussian Eliminations and one call of matrix inversion (each of which takes n^3 operations). Thus, the scaling between number of segments and the required time to simulate a run would be $3n^3$. Given that it takes about 8 minutes to simulate a run of the 10 segment lamprey on a SUN Ultra 100 (500 Mhz) workstation, it would take about 8 times longer to simulate a 20 segment lamprey (i.e. 64 minutes). To simulate a 40 segment version, it should take 64 times longer (about 8.5 hours) to complete a run. However, the time step needs to be reduced by a factor of 10 (i.e. h = 0.01 ms) for the simulation to be stable. This amounts to increasing the number of simulation steps (and simulation time) from 1 to 10. Thus, the simulation will require a total of approximately 3.5 days to complete. In terms of the 100 segment version, it will take 1000 times longer than the original version - in other words, a total of 8000 minutes. Again, the time step needs to be reduced by at least a factor of 10 (i.e. h = 0.001 ms) for the simulation to at least run. Thus, it would require nearly 2 months to complete one run of the 100 segment lamprey. This is simply not practical from the point of view of the availability of computing resources alone.

An ideal mechanical model of the lamprey should have 100 segments. The 10 segment model used in this thesis is a considerable simplification of that ideal (but computationally intractable) model. To check whether using the 10 segment model introduces any significant artefactual behaviour, I constructed 20 and 40 segment models and compared their performance with the 10 segment one. These models were created with body parameters similar to the 10 segment one. They are a step toward the 100 segment version yet they can be simulated within a reasonable amount of time, and they are able to give us an estimate of how the swimming performance of the lamprey scales in terms of the number of segments.

Table 3.2 lists details of the parameters for the 20 segment lamprey mechanical simulation. (The parameters for the 40 segment model are calculated similarily.) Note that to make a 20 segment model, each existing segment (in the 10 segment model) is divided into two pieces. This corresponds to reducing the segmental

length and mass by half. Since we want the acceleration of each link to be independent of the number of segments, muscle force is reduced by the same amount. The height of each link remains 30 mm. Similarily, in the 40 segment model, the length, mass and muscle force are four times less than in the original model. Note that in order to control the mechanical joints using the outputs of motoneurons from evenly spaced segments from the CPG, there are some differences in terms of the neural waves used to control the models.

link	w_i [mm]	l_i [mm]	m_i [g]	I_i [g mm^2]	$\lambda_\perp$ [Ns^2/m^2]	$\lambda_\parallel$ [Ns^2/m^2]
1	20.0	15.0	7.1	309.25	0.225	0.3
2	20.0	15.0	7.1	309.25	0.225	0.2
3	20.0	15.0	7.1	309.25	0.225	0.1
4	20.0	15.0	7.1	309.25	0.225	0.0
5	20.0	15.0	7.1	309.25	0.225	0.0
6	20.0	15.0	7.1	309.25	0.225	0.0
7	20.0	15.0	7.1	309.25	0.225	0.0
8	20.0	15.0	7.1	309.25	0.225	0.0
9	17.2	15.0	6.1	226.38	0.225	0.0
10	17.2	15.0	6.1	226.38	0.225	0.0
11	15.0	15.0	5.3	173.95	0.225	0.0
12	15.0	15.0	5.3	173.95	0.225	0.0
13	11.7	15.0	4.1	112.91	0.225	0.0
14	11.7	15.0	4.1	112.91	0.225	0.0
15	8.3	15.0	2.9	67.63	0.225	0.0
16	8.3	15.0	2.0	67.63	0.225	0.0
17	5.0	15.0	1.8	35.9	0.225	0.0
18	5.0	15.0	1.8	35.9	0.225	0.0
19	1.7	15.0	0.6	11.37	0.225	0.0
20	1.7	15.0	0.6	11.37	0.225	0.0

Table 3.2: Parameters for the mechanical simulations of a 20 segment model lamprey. w_i, i_i, m_i and I_i are the width, length, mass and inertia of link i respectively. $\lambda_\perp$ and $\lambda_\parallel$ are the λ factors used to calculate the water forces.

Given the amount of time required to simulate a 40 segment lamprey, I compared the swimming performance of the three models at only a few excitation combinations.

These data points were chosen to allow a comparison of the models at different levels of swimming efficiencies.

The results of the comparison are shown in Table 3.3. The table indicates that except at excitation combination (0.8, 20%), the number of segments does not make a significant difference in terms of swimming speed and efficiency. The reason for the relatively bigger drop in speed at this point is that the 10 segment lamprey swims with a larger lateral amplitude at the tail. As a result, this fish can propel forward better than one with more segments (although one with more segments is able to bend its body more smoothly). This appears only when the phase lag, speed and efficiency are very small. Since this thesis concerns evolving controllers of high speed and efficiency, and a 10 segment model requires much less time to simulate, it is acceptable to use it as the mechanical model of the lamprey.

Excitation	Amp	Freq [Hz]	Plag [%]	Mec period [s]	Mec λ [m]	Speed [m/s]	Efficiency
(0.4, 80%)	0.54	3.53	1.6	0.282	0.149	0.33	0.61
				0.282	0.146	0.31 [-6%]	0.59 [-3.3%]
				0.284	0.135	0.28 [-15.2%]	0.58 [-4.9%]
(0.4, 90%)	0.54	3.53	1.6	0.282	0.133	0.31	0.65
				0.281	0.143	0.30 [-3%]	0.60 [-7.7%]
				0.281	0.135	0.28 [-9.7%]	0.60 [-7.7%]
(0.4, 100%)	0.54	3.53	1.6	0.286	0.120	0.32	0.76
				0.286	0.140	0.31 [-3.1%]	0.63 [-17.1%]
				0.278	0.126	0.30 [-6.3%]	0.62 [-18.4%]
(0.5, 80%)	0.60	3.98	1.6	0.251	0.159	0.34	0.54
				0.252	0.157	0.32 [-5.9%]	0.51 [-5.6%]
				0.253	0.135	0.30 [-11.8%]	0.56 [3.7%]
(0.8, 20%)	0.75	4.83	0.7	0.207	0.203	0.22	0.22
				0.206	0.195	0.14 [-36%]	0.15 [-31.8%]
				0.207	0.125	0.13 [-41%]	0.21 [-4.5%]
(0.8, 60%)	0.73	5.13	1.5	0.195	0.157	0.40	0.50
				0.189	0.175	0.44 [10%]	0.48 [-4%]
				0.189	0.166	0.38 [-5%]	0.43 [-14%]

Table 3.3: Comparison of neural and mechanical parameters of the 10, 20, 40 lamprey models. Note that each table element contains three rows. The top row contains data for the 10 segment model while the second and third row contain data for the 20 and 40 segment model respectively. Numbers in brackets represent the percentage change in speed/efficiency from that achieved by the 10 segment lamprey model.

Despite the differences discussed above, my implementation of the model lamprey, like those of Ijspeert and Ekeberg, can produce anguilliform swimming very similar to the real one, and is thus a valid mechanical model for the purposes of this thesis.

3.9 Summary

In this chapter, I presented my reimplementation of Ekeberg's neural model and Ijspeert's mechanical lamprey model. Quantitative analysis of my implementations of the Ekeberg models show that the neural model is capable of producing several characteristics found in the real lamprey: 1) an isolated segmental network can be made to oscillate, 2) the frequency of oscillation increases with global excitation from the brainstem, 3) the phase lag can be varied independently by changing the extra excitation in the five most rostral segments, 4) the phase lag is fairly constant along the entire swimming CPG.

[Ijspeert, 1998] corrected some of the parameters in Ekeberg's mechanical model. I used Ijspeert's parameters because they are correct.

A new speed calculation algorithm is proposed followed by a step-by-step description of the procedure for plotting the neural characteristics (amplitude, frequency and phase lag) and speed surfaces.

When the outputs of the CPG are used to control the model lamprey, the artificial fish exhibits anguilliform swimming. The frequency of oscillation and phase lag can be varied almost independently by changing the global excitation and extra excitation respectively. The swimming speed can be modulated by changing the level of excitations from the brainstem.

Finally, although there are some differences in implementation details, comparison of the frequency, phase lag and speed surfaces obtained by me (same programs used in Or and Hallam [2000]) and Ijspeert [1998] has shown that our lamprey systems are similar. Unfortunately, Ekeberg [1993] does not provide enough details for comparisons. The differences in implementation details among the three of us are listed in Tables 3.4 and 3.5.

	Scheme to connect the neurons	Integration method and time step
Ekeberg	Postsynaptic normalization	Euler with 10 ms
Ijspeert	Postsynaptic normalization	Fourth order RK with 5 ms
Or	Postsynaptic normalization	Fourth order RK with 1 ms

Table 3.4: Comparison of Ekeberg's neural model implemented by Ekeberg, Ijspeert and Or.

	Means to connect the motoneuron outputs to the muscles	Parameters for mechanical model	Integration method and time step
Ekeberg	One segment to a joint	Few bugs in mass, muscle-torque and water force equations	Euler with 1 ms
Ijspeert	Same as Ekeberg	Same as Ekeberg with bugs fixed	4th orde RK with 0.5 ms
Or	Average of 10 segments to a joint	Same as Ijspeert	4th order RK with 1 ms

Table 3.5: Comparison of Ekeberg's mechanical model implemented by Ekeberg, Ijspeert and Or.

Chapter 4

A study on the robustness of the lamprey swimming controllers

In his excellent book on fish swimming, Videler [1993] mentions that:

> Swimming speed limits and endurance are directly related to food capture, escape from predators and reproduction. Therefore they are presumed to be subjected to strong selection pressures that enhance evolutionary fitness. This Darwinian fitness requires an individual not only to survive but also to produce fertile offspring. Against this background, all aspects of swimming performance are potentially crucial. The maximum speed a fish can sustain indefinitely, the endurance at higher speeds and the absolute maximum allowed burst speed are of high ecological importance and therefore important to investigate. It is also crucial to know how much time and energy must be allocated to swimming in order to achieve the highest fitness and how this affects the energy budget of a fish. (p. 207)

This chapter investigates the effect of scaling body parameters on the maximum speed. In Chapter 6, I investigate the energy aspect of swimming in terms of efficiency. Since the robustness in maximum swimming speed against changes in body scales is used as an evaluation criterion, the effect of scaling the model lamprey's body parameters is described. Then, a preliminary study on the robustness of a few existing lamprey swimming controllers is presented. In the study, neuromechanical simulations of lampreys with variable body configurations and controllers are conducted. Performance of the controllers is assessed by comparing their maximum achievable speed. Results indicate that the biological controller hand-crafted by Ekeberg [1993] and controller 2 of

Ijspeert [1998] behave very differently. This leads to a series of investigations at both behavioral and neural levels to explain the differences. As part of the investigation, a study characterizing the mechanical lamprey model using a sinusoidal controller is conducted. Throughout the investigations, I found that the mechanical lamprey model I am using exhibits multiple resonances (Chapter 5). It is this property and the difference in neural organization which cause the controllers to behave differently.

4.1 Motivation

Recently, there has been a growing interest in building biologically inspired swimming robots [Ayers, 1995, Burdick and Ostrowski, 1998, Kato and Inaba, 1998, McIsaac and Ostrowski, 1999, Sfakiotakis et al., 1999, Ayers et al., 2000, Knutsen, 2000]. This is because robots which can swim like a fish are highly desirable in several application areas - they can be used for underwater exploration and data collection, undisturbed monitoring of marine animals, surveillance, film making, etc.

Over the course of evolution, the neural and mechanical systems of fish have evolved to allow efficient swimming. For instance, when fish swim, the propulsion is noiseless and highly energy efficient [Sfakiotakis et al., 1999]. Morever, fish can maneuver through small gaps with ease. Such ability allows them to enter places no underwater vehicle has ever gone before. Hence, by studying the neural and mechanical systems responsible for fish swimming, we may learn how to build autonomous underwater vehicles (AUV) with propulsors of possibly higher performance and dexterity than those available at present.

Currently, several obstacles have to be overcome in order to build a swimming robot with at least the same efficiency as its biological counterpart. Some of these obstacles are power, actuators and an appropriate control architecture. Due to the high cost of developing the mechanical system, it is necessary to turn to computer simulations as a first step in order to find a controller which is the most suitable for controlling a mechanical body subject to physical variations due to craftsmanship or design. Since we are not able to simulate all aspects of the real world precisely, a controller which can tolerate simulation infidelity is also very important.

Given the imperfections of electronics and mechanical parts, it is necessary to know how tolerant the controllers are to these changes. This section provides a few definitions of robustness and discusses why they are important and how they can be tested. The following list summarizes the types of robustness which I am going to discuss:

1. Robustness against variation of body parameters.
2. Robustness against noise in connection strengths among the neurons.
3. Robustness against variations in neural parameters.
4. Robustness against changes in the input signals.

4.1.1 Robustness against variation of body parameters

Although there is some literature on the scaling effects of fish, there is no data on how the speed varies with body scales in the case of the lamprey. Although Wu [1977] provides an introduction to the scaling of fish, his focus is on carangiform and lunate-tail (rather than anguilliform/undulatory) locomotion. Holwill [1977] mentions undulatory propulsion in organisms of different sizes but the focus is on organisms with low Reynolds numbers[1] such as bacteria and small worms. Wardle [1977] provides a good introduction to the effects of size on the swimming speed of fish. He predicted the maximum swimming speed for all fish at any temperature. From his maximum burst swimming speed vs. fish length graph, fish of length 0.3 m (a typical size for an adult lamprey) have a maximum velocity of about 0.4 m/s which is a bit lower that achieved by the real lamprey which is about 2 body lengths/sec (i.e. 0.6 m/s for a 0.3 m lamprey) [Petersen, 2001].

Robustness against variation in body parameters is very important from an adaptive behavior viewpoint because as an organism grows, its body characteristics change. A locomotion controller which can tolerate such variation is essential to the survival

[1]Reynolds number is defined as the ratio between inertial and viscous forces [Schmidt-Nielsen, 1984]. At low Reynolds number, viscous forces dominate and the water flow is laminar. In other words, the velocity of the water at any point is constant [Bloomfield, 1997].

of the being. To test this type of robustness, we can either systematically or non-systematically change the scaling of different body parameters and then observe how this affects the maximum swimming speed.[2]

Or and Hallam [2000] studied the effect that scaling had on the maximum swimming speed of several lamprey models. We found that the controllers under investigation behaved differently. In particular, as the scale increased, one of the controllers (the evolved controller 2 to be described in Section 4.4) increased its frequency of oscillation. However, the other controllers (the biological controller proposed by Ekeberg in particular) decreased the oscillation frequency. In Or et al. [2002], we investigated the potential for evolving robust swimming controllers using Genetic Algorithms. Although the best evolved controller is more robust (has less change in speed at different scales) than the controllers investigated in Or and Hallam [2000], only 25% of the runs produce interesting results in about three months. In Chapter 6, through studying the evolution of efficient swimming controllers, I find that the fitness function for evolving efficient controllers indirectly brings in the robustness property. It also takes less time to evolve controllers which have high swimming efficiency and good robustness in swimming speed.

4.1.2 Robustness against noise in connection strengths among the neurons

Hellgren et al. [1992] used a detailed model of the lamprey segmental network (described in Ekeberg et al. [1991] and Wallén et al. [1992]) to study how variation in cell and synaptic parameters affects burst pattern.[3] They found that the robustness of the burst patterns as well as the working frequency range can be increased by incorporating a random variation in certain cells (such as duplicating the number of interneurons per population within the network) and synaptic parameters. Furthermore, the popula-

[2]Refer to Subsection 3.5.3 for the reasons for using maximum speed as the basis of comparison of fish performance.

[3]According to Hellgren et al. [1992], the model consists of a population of interneurons whose cellular properties such as size and membrane conductance (including voltage dependent ion channels) are taken into consideration. The model is detailed enough to take all experimentally established important neural properties into consideration, yet it is simple enough to accomodate the simulation of several neurons.

tion model they used is capable of producing stable burst activity over a large range of oscillation frequencies even without the LIN neurons.

Later, Ferrar et al. [1993] studied the robustness of the lamprey segmental network proposed by Buchanan and Grillner [1987]. Unlike the one investigated by Hellgren et al. [1992], the neurons in this model are less detailed and each of them represents the entire populations of neurons of the same type. Ferrar and his colleagues observed how the neural characteristics of the network (such as cycle duration, burst proportion and left-right phase lag) change when it is subjected to two different noise conditions: 1) long time scale noise at synaptic connections from the driver cells (brainstem in Ekeberg model) to all the other cells and 2) short time scale noise at all synaptic connections. The synaptic strengths of the former test varied from 50 to 100% only at the beginning of the simulations. In the other test, the synaptic strengths varied from 0 to 200% at each time step. Ferrar and colleagues found that the network is less affected by noise in the short time scale because the noise effect tends to average out over a cycle. Furthermore, they found that increasing the number of neurons in the network allows an increase in robustness despite the influence of noise. It would be interesting to investigate how noise in both segmental and intersegmental connections affect the neural characteristics and swimming performance of Ekeberg's biological model as well as some of the evolved CPGs. One way to test this is to vary the interneuronal connections randomly (refer to Chapter 7).

4.1.3 Robustness against variation in neural parameters

If the properties of the neuron change, the network may stop oscillating. The range of oscillation frequency can also change. One way to test this is to observe how changing the neuron parameters affects the amplitude, frequency, phase lag and speed surfaces. Hellgren et al. [1992] found that their segmental network (modelled using the population model with cell size, membrance conductance, synaptic delay and conductance randomly distributed around experimentally obtained mean values) exhibits a larger working frequency range and is more robust against background noise than networks with cells and synaptic parameters chosen without variation (i.e. equivalent to a model with one cell per population). Ferrar et al. [1993] did a similar study. They investi-

gated the effects of duplicating the numbers of neurons (of types CIN, EIN and LIN) against noise in the segmental network model proposed in [Buchanan and Grillner, 1987]. They found that networks with increasing numbers of neurons of each type are more stable (e.g. show lower variability in the standard deviation of cycle duration) than networks with only one neuron per population. The results from these studies conclude that it is beneficial to use networks with more neurons of each type as random variation in the CPG is less disturbing.

4.1.4 Robustness against changes in input signals

In order to investigate how swimming controllers deal with continuously changing input commands, Ijspeert [1998] added a simple visual system on top of his evolved controllers. The controllers were asked to track a randomly moving object. Results show that despite the periodic movements of the head and random movement of the target, the best controller is capable of tracking the moving object. Later, Ijspeert [1998] tested whether his controller was able to cross the speed barrier with a fixed target located on the opposite side of the barrier. He found that without sensory feedback and correction of input commands from the visual system,[4] the lamprey was able to cross a speed barrier that had a speed of 90% of the lamprey's swimming speed. However, with the tracking behavior, the speed of the barrier could be increased to 95%. The speed barrier was further increased to 140% when evolved sensory feedback was incorporated in the locomotion system. His results show that the evolved controllers are robust against changes in input signals. Thus, they can integrate higher command systems such as the vestibular system for orientation to exhibit more complicated behaviors.

Turning is induced by differences in inputs from the brainstem to the left and right neurons. A controller which cannot tolerate noise from the input signals can cause unnecessary turnings which prevent the lamprey from reaching its goal. A simple way to test such *controllability* is to use the visual system to track a randomly moving target. Ijspeert showed that swimming gait is very robust against variation in input commands

[4]The speed barrier tends to cause the lamprey to change swimming direction. On the other hand, the visual system helps the lamprey to keep its swimming direction towards the target by continuing to send input signals to the CPG to keep the lamprey on track.

(unpublished work). His lamprey is able to track a randomly moving target despite noise (such as periodic head movements and random appearance of the target) in the CPG.

4.2 Types of robustness to be investigated in this dissertation

A few researchers have studied the robustness of model lamprey CPGs, some at the segmental level [Hellgren et al., 1992, Ferrar et al., 1993], others at the intersegmental level [Wadden et al., 1997, Ijspeert, 1998, Or and Hallam, 2000].

Given that a real robotic lamprey is unlikely to match the simulated one in detail and that this is one of the most important of the kinds of robustness discussed above, I chose to study robustness against variation in body parameters. Inspired by Ekeberg [1993], Ijspeert [1998] used GA to evolve 2D swimming controllers similar to the one hand-crafted by Ekeberg, but which are able to produce faster swimming. In this chapter, I investigate the robustness of two of these controllers with respect to systematic variation of body parameters such as mass, body length and inertia.[5] In Chapter 7, I study the robustness of the swimming controllers against noise in neural connections.

4.3 The effect of scaling body parameters

The experiments which I am going to describe require the scaling of different body parameters. This section provides background information on how different body parameters in the model are scaled.

Body parameters are inter-related. Length scales linearly with the scale factor, which varies from 70% to 150% of the original.

[5]The material in the rest of this chapter is based closely on [Or and Hallam, 2000]. For the investigation on using the GA to evolve controllers which are robust against variation in body scales, refer to [Or et al., 2002].

To calculate the body mass of a mechanical link, its volume is required. Since each link has an elliptical cross-section, the volume $[mm^3]$ can be calculated as:

$$Volume = \pi(\frac{width}{2})(\frac{height}{2}) \cdot length$$

The mass $[g]$ is then:

$$Mass = \rho \cdot Volume$$

where ρ is the lamprey density (assumed to be equal to water density of 1000 kg/m^3 in these experiments).

In order to keep the accelerations of body links similar between lampreys of different sizes, the forces generated by the muscles are scaled to match the changed mass (personal communication with Hallam and Ijspeert). This corresponds to the assumption that the force generated by muscles scales with volume rather than cross-section as observed in real animals [Fitzgerald, 2002]. Note that this scaling amounts to a change in the low frequency gain of the muscle system with no change in the musculature's frequency response. Thus, when the body segment is scaled in all three dimensions, the mass and the muscle force are scaled as the cube of the scaling factor. Since the moment of inertia $[gmm^2]$ of an elliptical cylinder along the axis perpendicular to the x, y axes is

$$Inertia = Mass \cdot (\frac{width^2}{16} + \frac{length^2}{12}),$$

the inertia is scaled as the fifth power.

The water forces $F_{water||}$ and $F_{water\perp}$ depend on the parallel and perpendicular areas of the lamprey segment respectively (Subsection 2.5.4). Hence, each water force is scaled as the second power.

4.4 Method

The following four controllers are under investigation.

1. *The Biological Controller*, hand-crafted by Ekeberg [1993] based on physiological data.

2. *Controller 2*, evolved by Ijspeert et al. [1999] using Ekeberg's segmental oscillator and intersegmental couplings evolved by GA.

3. *Controller 3*, evolved by Ijspeert et al. [1999], with both intra- and intersegmental connections evolved by GA.

4. *A Sinusoidal Controller*, in which motoneuron outputs are sinusoidal pulses generated analytically for any given frequency and intersegmental phase lag, included for comparison purposes (see below).

Controllers 2 and 3 were evolved by Ijspeert using staged evolution (see Section 2.7). They correspond to the ones obtained in RUN2 and RUN7 mentioned in [Ijspeert et al., 1999]. The reason why these two controllers are chosen is that they are capable of producing a wide range of speeds within the excitation ranges of the biological one. This allows better comparison among the controllers. Table 4.1 shows the connection weight matrix for the biological and the two evolved controllers respectively (they are hereby called the *CPG controllers*). The amplitude, frequency, phase lag and speed surfaces for these three controllers are shown in Figure 4.1, Figure 4.2 and Figure 4.3 respectively. (For the procedure used to generate these surfaces, refer to Section 3.6.)

For comparison purposes, the fourth controller (*standard sinusoidal*), which generates sinusoidal pulses with controllable frequency and phase lag, was used. The output amplitude for this controller is standardized to 0.8 (maximum motoneuron output is 1; the other controllers typically produce motoneuron amplitudes of 0.5 to 0.7). Ijspeert (personal communication) used a similar approach when characterizing his mechanical simulator. Note that later in this chapter I use another type of sinusoidal controller called the *matched sinusoidal controller*, whose maximum amplitude is adjustable. Unless otherwise stated, the standard controller is used by default.

From	To	Biological Controller	Controller 2	Controller 3
EINl	MNl	1 [5, 5]	1 [4, 1]	-0.4 [9, 2]
	EINl	0.4 [2,2]	0.4 [5,3]	-0.8 [12, 4]
	LINl	13 [5,5]	13 [4, 1]	-
	CINl	3 [2, 2]	3 [11, 2]	0 [12, 8]
	MNr	-	-	-
	EINr	-	-	-0.9 [5, 10]
	LINr	-	-	-
	CINr	-	-	-3.5 [2, 2]
LINl	MNl	-	-	-
	EINl	-	-	-
	LINl	-	-	-
	CINl	-1 [5, 5]	-1 [3, 9]	-
CINl	MNl	-	-	-3.2 [8, 1]
	EINl	-	-	-3.8 [12, 10]
	LINl	-	-	-
	CINl	-	-	-
	MNr	-2 [5, 5]	-2 [11, 11]	-
	EINr	-2 [1, 10]	-2 [1, 0]	-0.7 [1, 10]
	LINr	-1 [1, 10]	-1 [9, 4]	-
	CINr	-2 [1, 10]	-2 [0, 5]	-3.7 [9, 9]
BS	MNl	5	5	3.8
	EINl	2	2	0.8
	LINl	5	5	0
	CINl	7	7	13.6

Table 4.1: Connection weight matrix for the biological and the two evolved controllers. Positive numbers mean excitatory connections while negative numbers mean inhibitory connections. Left and right neurons are indicated by *l* and *r*. Due to symmetry between the left and right, the connection weights of the neurons on the right are not shown here. BS stands for the brainstem. The number of extensions to the neighboring segments are shown in the brackets. The first entry represents extension in the rostral direction while the second entry represents extension in the caudal direction.

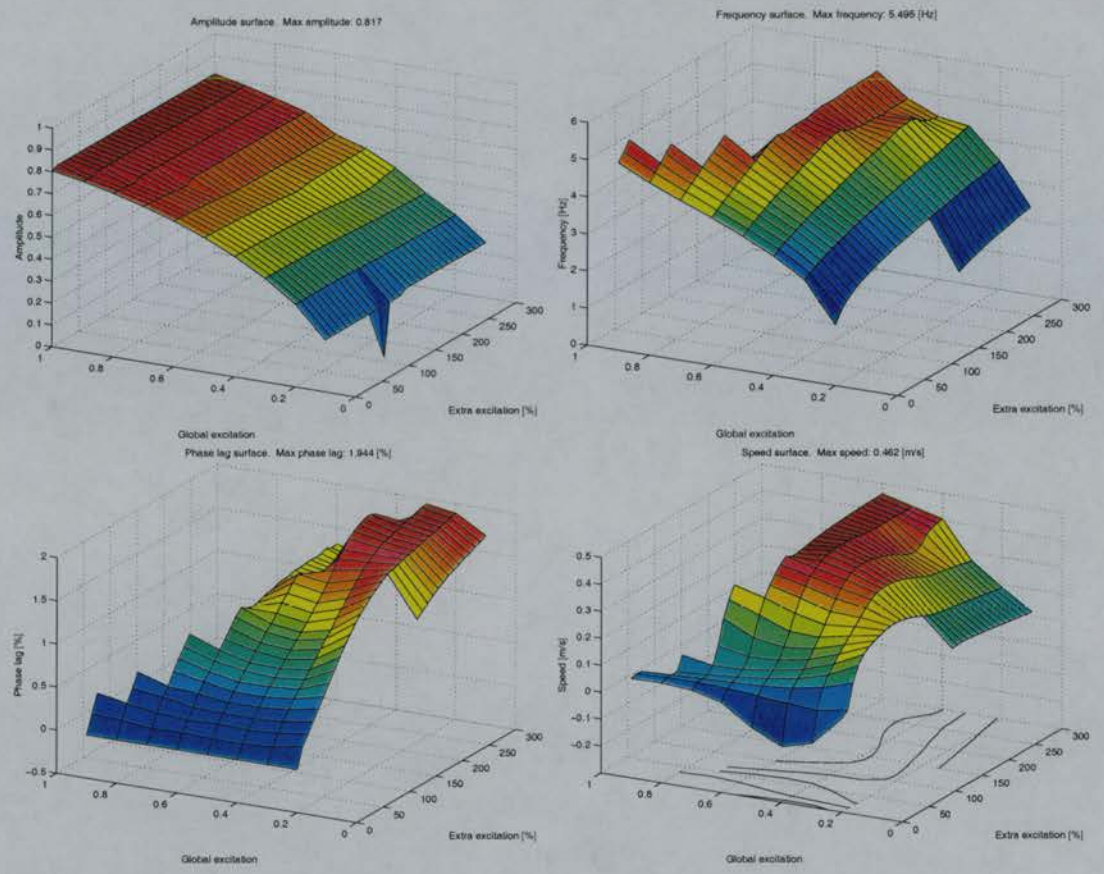


Figure 4.1: Amplitude, frequency, phase lag and speed surfaces for the biological controller

4.4.1 Description of Experiments

To test how robust the controllers are, I conducted the following experiments on each of them and then compared their performance in terms of maximum achievable swimming speed. In each experiment, the scale factor varies from 0.7x to 1.5x. Since the controllers can generate any speed up to their maximum by varying the excitation levels, maximum speed is a reasonable measure of performance. For a controller to be robust, its maximum achievable speed should vary little as body parameters vary. For each successive experiment, the number of independent variables is increased by one.

- *Experiment 1: Varying the body density alone.* The mass (m_i) of each body segment is scaled over the range from -30% to +50% with no change in body dimensions. Each corresponding segmental inertia (I_i) which is directly propor-

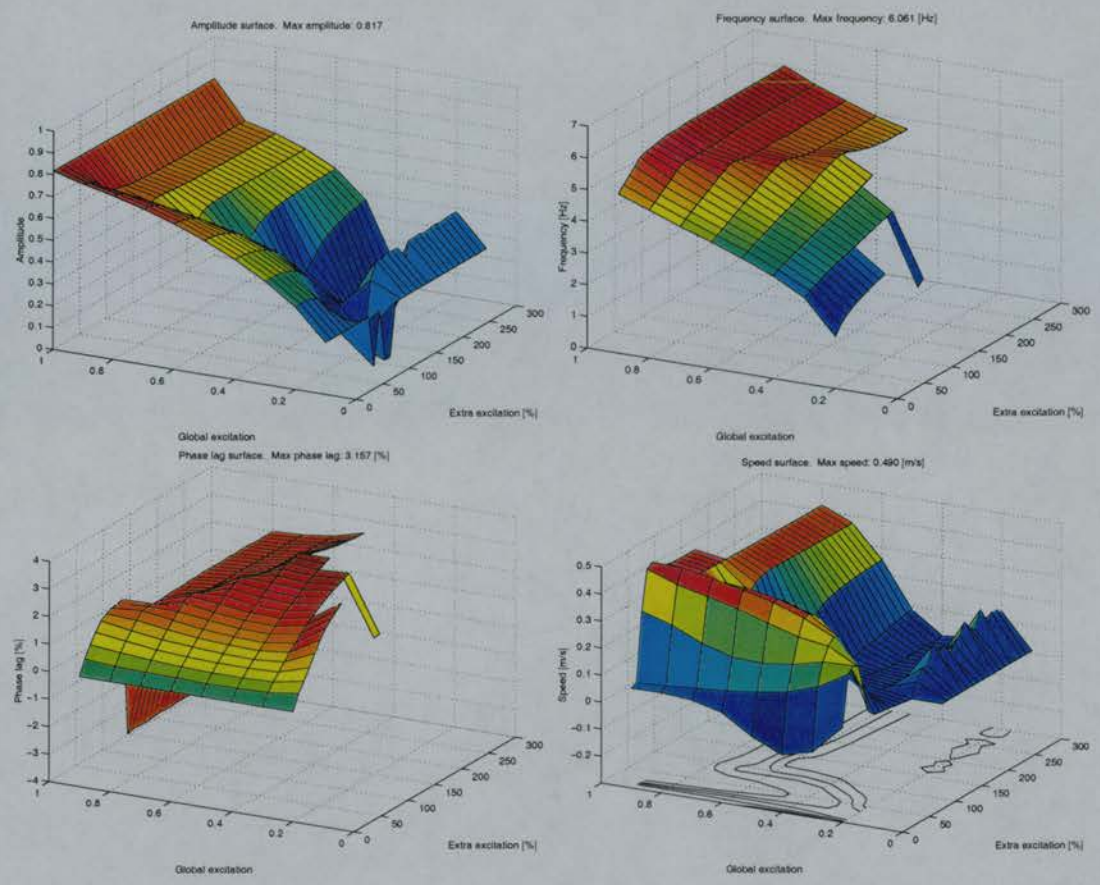


Figure 4.2: Amplitude, frequency, phase lag and speed surfaces for controller 2

tional to the mass is also scaled.

- *Experiment 2: Varying the body mass and muscle force.* The masses and inertias are scaled as in Experiment 1 without change of body dimensions. In addition, the muscle forces are scaled to match the changed mass. This corresponds to an assumption that the force generated by muscle scales with its volume rather than its cross-section (recall that $mass = \rho \cdot Volume$).

- *Experiment 3: Varying the body length.* The body lengths (but not the cross-sections) are scaled over the range from -30% to +50% of the normal size. This scales mass, inertia and muscle forces linearly with length (i.e. the lamprey's

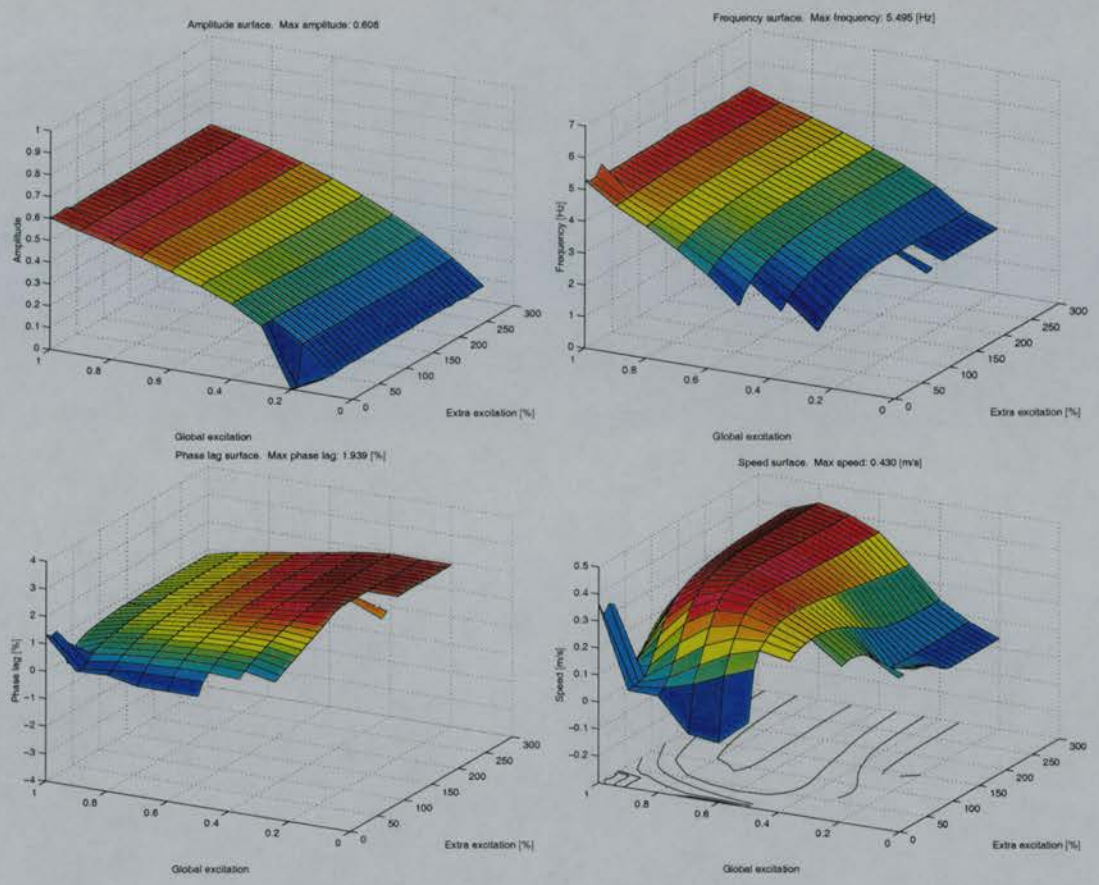


Figure 4.3: Amplitude, frequency, phase lag and speed surfaces for controller 3

density is assumed to be the same as that of water).[6] Water forces perpendicular to the body ($\lambda_\perp$) scale linearly because the cross-section of the segment for sideways motion changes with the length.

- *Experiment 4: Varying the entire body.* Each body segment is scaled in all three dimensions. This scales mass and muscle force as the cube of the scaling factor, inertia as the fifth power, and water forces as the square. Again, the lamprey density is assumed to be the same as that of water. (Note that, with the scalings investigated, this results in a wide variation of the body. For instance, the mass changes from around 34% to 337.5% of its normal value).

[6] Recall $mass = \rho \cdot Volume$, where $Volume$ = cross-section area$\cdot length$. Also, the muscle forces scale with volume rather than the cross-section.

4.4.2 Procedures

Each of the experiments mentioned above was conducted with the following stages. Note that the procedure is based on that described in Section 3.6:

- *Stage 1: Neuromechanical simulations.*
  The procedure described in Stage 2 in Section 3.6 is used here. The difference is that the global excitation ranges from 0.0 to 1.0 in steps of 0.1 while the extra excitation ranges from 0% to 300% in steps of 25%.

- *Stage 2: Speed calculation.*
  The first speed calculation method mentioned in Section 3.5 is used here to calculate the swimming speed.[7]

- *Stage 3: Comparative analysis of swimming controllers within the same experiment.*
  Using the results from Stage 2, for each experiment, the set of controller parameters that resulted in the fastest swimming speed is determined for each body scale. The corresponding maximum speed is recorded.

- *Stage 4: Comparison of neural and programmed controllers.*

The result of Stage 3 is a set of neural parameters (global and extra excitation) that result in the maximum swimming speed for each scale. To investigate the extent to which the performance depends on the CPG output pulse shape, control runs of the sinusoidal controller with matched control parameters (amplitude, frequency and phase lag) are performed (Subsection 4.5.4).

4.5 Results and Discussion

The four graphs in Figure 4.4 compare robustness of all four controllers within the same experiments. Figure 4.5 compares their robustness across different experiments.

[7]The reason for using the simpler method is that this set of experiments was carried out before the more accurate speed calculation algorithm was devised.

The maximum speed achieved by the controllers with the original body configuration is shown in Table 4.2.

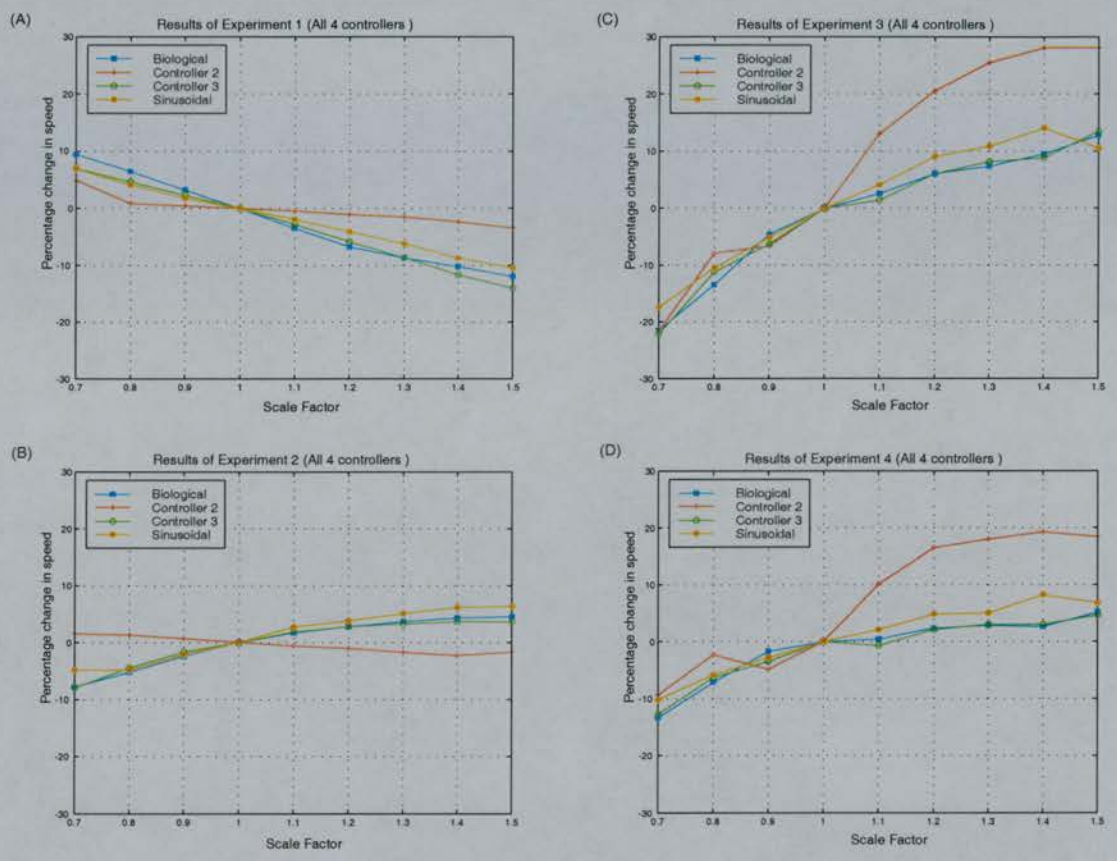


Figure 4.4: Comparison of speed achieved by the four controllers in each experiment. Values on the x-axis represent multiplication factors of body parameters under investigation. (From Or and Hallam [2000].)

4.5.1 Results of each experiment

Experiment 1: Varying the body mass.

Figure 4.4A shows how variation in body mass affects the swimming speed. For all four controllers, as the body mass increases, the swimming speed decreases. The speed change of each controller with respect to the nominal speed reported in Table 4.2 is shown in Table 4.3.

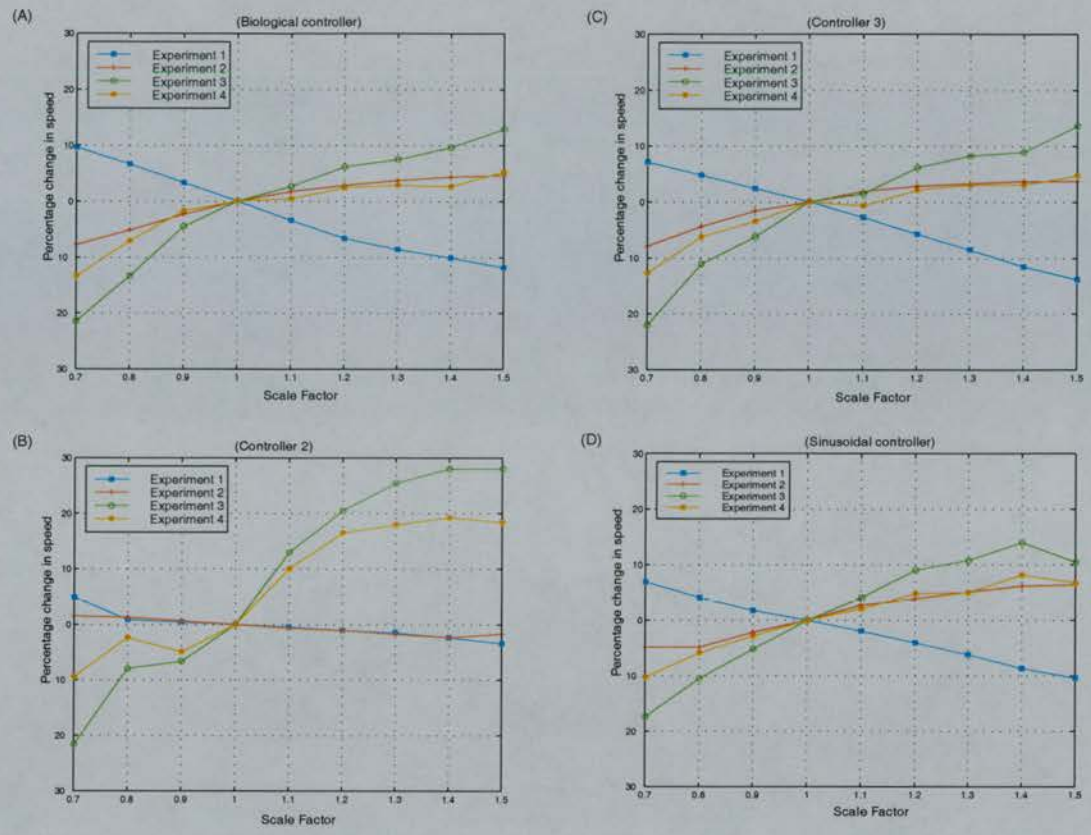


Figure 4.5: Comparison of speed achieved by the four controllers across different experiments. Values on the x-axis represent multiplication factors of body parameters under investigation. (From Or and Hallam [2000].)

Experiment 2: Varying the body mass and muscle force.

Figure 4.4B shows that, except in controller 2, as the mass and the muscle force increase together, the maximum swimming speed increases as well. The speed change of each controller is shown in Table 4.4.

Controller	Speed [m/s]
Biological	0.462
Controller 2	0.467
Controller 3	0.430
Standard sinusoidal	0.563

Table 4.2: Speed achieved by different controllers with the original body configuration.

Controller	Speed [m/s] (0.7x)	Speed [m/s] (1.5x)	Difference in speed [%]
Biological	0.506	0.407	9.52, -11.9
Controller 2	0.490	0.451	4.93, -3.43
Controller 3	0.460	0.371	6.98, -14.0
Standard sinusoidal	0.602	0.504	6.93, -10.5

Table 4.3: Speed achieved by different controllers in Experiment 1.

Controller	Speed [m/s] (0.7x)	Speed [m/s] (1.5x)	Difference in speed [%]
Biological	0.426	0.483	-7.79, 4.55
Controller 2	0.474	0.459	1.50, -1.71
Controller 3	0.394	0.444	-7.94, 3.64
Standard sinusoidal	0.528	0.590	-4.86, 6.30

Table 4.4: Speed achieved by different controllers in Experiment 2.

Experiment 3: Varying the body length.

Figure 4.4C shows how change in body length affects the swimming speed. The results show that as the body length is increased, the swimming speed increases as well. Furthermore, for each controller, the amount of increase is more than that in Experiment 2, with a particularly large increase in the performance of controller 2 for the larger, heavier bodies. The speed change of each controller is shown in Table 4.5.

Controller	Speed [m/s] (0.7x)	Speed [m/s] (1.5x)	Difference in speed [%]
Biological	0.363	0.521	-21.4, 12.8
Controller 2	0.366	0.598	-21.6, 28.1
Controller 3	0.335	0.488	-22.1, 13.5
Standard sinusoidal	0.465	0.622	-17.4, 10.5

Table 4.5: Speed achieved by different controllers in Experiment 3.

Experiment 4: Varying the entire body.

Figure 4.4D shows how scaling the entire body affects the swimming speed. The results indicate that as the body size increases, the swimming speed increases accordingly. The changes in speed are less than those in Experiment 3, and controller 2 still performs much better than the rest at increased body dimensions. The speed change of each controller is shown in Table 4.6.

Controller	Speed [m/s] (0.7x)	Speed [m/s] (1.5x)	Difference in speed [%]
Biological	0.400	0.486	-13.4, 5.20
Controller 2	0.423	0.553	-9.40, 18.4
Controller 3	0.375	0.450	-12.8, 4.65
Standard sinusoidal	0.505	0.600	-10.3, 6.73

Table 4.6: Speed achieved by different controllers in Experiment 4.

4.5.2 Discussion of individual experiments

The results from each individual experiment are encouraging. Most of the controllers show variation in performance as the body parameters are varied. The range of variation is less than the variation of the body parameters. This is good because if variations in performance were too great when there is only a small change in body scale, then the controllers would not be robust. However, it is also reasonable to accept that there is some change in swimming performance when the body parameters are scaled.[8]

All controllers swim more slowly as the body mass alone is increased. This is because the heavier body, with unchanged muscle force, has a larger time constant so that maximum speed is achieved at a lower frequency of the CPG.[9] This results in slower swimming. When the muscle force is scaled with the mass, all the controllers except controller 2 show an increase in performance when the body mass increases. In other words, the increased muscle force allows each controller (except controller 2) to keep up the swimming speed despite its heavier body. The speed curve of controller 2 in this experiment remains similar to the curve it had in Experiment 1, showing that controller 2 has already tried its best to keep up the swimming speed in spite of an increase in body mass (i.e. the amplitudes of the motoneuron outputs from controller 2 are quite high already even when the muscle force is not scaled in Experiment 1).[10]

In practice, a real robotic lamprey can be built to have buoyancy which matches that of the water. Experiments 3 and 4 investigate the effects of scaling under this condition. All controllers show increasing performance as the body is scaled upwards. The main difference between the two experiments is that in Experiment 4, all body dimensions scale upwards and this results in increased drag as the frontal and lateral cross-sections of the body become larger (refer to Section 4.3 for further information on the scaling effect).

Controller 2 exhibits the largest increase in performance as all body dimensions are

[8] It would be interesting to compare the swimming speeds of the simulated lamprey with those of baby and adult lampreys. Unfortunately, nobody has studied the swimming speeds of the real lamprey throughtout development.

[9] Consider the body as a mass-spring system. The frequency of oscillation is then defined as $f = \frac{1}{2\pi}\sqrt{\frac{k}{m}}$ where k is the spring constant and m is the mass.

[10] In Section 5.3, I will show that the neural organization of controller 2 allows it to accept more excitations (and hence produce higher muscle force than other controllers) before failing to oscillate.

increased. This is not caused by an increase in mass and/or muscle forces (compare Experiments 3 and 4 with Experiments 1 and 2). It is not clear why this controller is so much better at driving the larger bodies. A comparison with the matched sinusoidal controller (refer to Subsection 4.5.4) shows that all CPG controllers achieve consistently higher swimming speed than the sinusoidal controller and the latter exhibits less variation in performance than the former.

The standard sinusoidal controller achieves a higher maximum speed than the CPG controllers (refer to Table 4.3 to 4.6). This is because this controller has a larger motoneuron amplitude of 0.8 which results in more powerful body movements. If the sinusoidal controller's output amplitude were closer to that of the others, its performance would match theirs (refer to Subsection 4.5.4).

4.5.3 Comparison of controllers across different experiments

Figure 4.5 allows comparative analysis of the performance of each controller across the four experiments. Interestingly, the pattern of variation for controller 2 is very different from that of the others (which have similar scaling behavior in all the experiments). Section 5.1 onwards investigate the unusual behavior of this controller.

4.5.4 Comparison with the matched sinusoidal controller

As mentioned at the end of Subsection 4.4.2, the performance of the CPG controllers (the biological controller, controller 2 and controller 3) are compared with that of the matched sinusoidal controller. By matching, I mean that the amplitude, frequency and phase lag of the control signals are chosen to correspond to the parameters for the CPG controllers. The equations for the sinusoidal controller are shown as follows:

$$T = \frac{1}{frequency} \cdot 1000$$

$$\omega = \frac{2 \cdot \pi}{T}$$

$$lag = \frac{(seg+1)\cdot plag \cdot T}{100}$$

$$phase = \omega \cdot (t+1) - \omega \cdot lag$$

$$mn = \alpha \cdot sin(phase)$$

$$MN_l(t,seg) = \begin{cases} mn & \text{if positive} \\ 0 & \text{otherwise} \end{cases}$$

$$MN_r(t,seg) = \begin{cases} -mn & \text{if negative} \\ 0 & \text{otherwise} \end{cases}$$

where T is the period and α is the amplitude. $MN_l(t,seg)$ and $MN_r(t,seg)$ are, respectively, the outputs of left and right motoneurons in segment $seg \in [0,99]$ at time $t \in [0,7999]$ ms.

To summarize: across all four experiments and scalings under investigation, the CPG controllers exhibit faster swimming than their matched sinusoidal counterparts in 100 out of 108 cases.[11] The speed difference in the remaining 8 cases is small (typically 3 to 10 mm/s) while in the majority of cases the CPG controller is about 15-30 mm/s faster and in some cases even faster. The matched sinusoidal controller is particularly poor at controlling the lighter bodies compared to the biological controller. In Experiments 3 and 4, the biological controller achieved a swimming speed of about 230 mm/s faster than the matched sinusoidal controller. Detailed results of comparisons are presented in Table 4.7 for controller 2 in Experiments 3 and 4.

To find out why there is a performance difference between the CPG controller and the matched sinusoidal controller, I investigated the control signals of the two controllers at the scale 1.1x of Experiment 4, where there was a difference in speed, and therefore the differences in control pulse characteristics would be obvious. First I considered whether the reason could be that the averaged MN activity driving the first joint

[11] Four experiments, three controllers at nine different scales.

	Experiment 3			Experiment 4		
Scale Factor	Speed of Controller 2 [m/s]	Speed of Sinusoidal Controller [m/s]	Difference [m/s]	Speed of Controller 2 [m/s]	Speed of Sinusoidal Controller [m/s]	Difference [m/s]
0.7	0.366	0.377	-0.011	0.423	0.423	0.000
0.8	0.430	0.420	0.010	0.456	0.441	0.015
0.9	0.436	0.420	0.016	0.444	0.428	0.016
1.0	0.467	0.462	0.005	0.467	0.462	0.005
1.1	0.528	0.510	0.018	0.514	0.498	0.016
1.2	0.563	0.551	0.012	0.544	0.527	0.017
1.3	0.586	0.563	0.023	0.551	0.535	0.016
1.4	0.598	0.560	0.038	0.557	0.529	0.028
1.5	0.598	0.511	0.087	0.553	0.516	0.037

Table 4.7: Comparison of the scaling performance of controller 2 with a matched sinusoidal controller in Experiments 3 and 4

of controller 2 had a higher amplitude than the corresponding one from the sinusoidal controller.[12] If this hypothesis were correct, increasing the output (which drives the first joint) from the sinusoidal controller should allow the lamprey to achieve a speed similar to what it would achieve if it were being driven by controller 2. However, experimental results showed that the speed does not increase. Hence, the higher amplitude in itself is not responsible for the higher speed achieved by controller 2. The difference in performance should therefore be caused by the difference in pulse shape. The results shown in Figure 4.6 were obtained when I superimposed the averaged MN activity from the matched sinusoidal controller to that of controller 2. The figure shows that when the pulses from both controllers are active, the ones from controller 2 have a higher overall amplitude at each time instance. This means that at each instance, controller 2 can exert a larger force than the sinusoidal controller, thus allowing the model lamprey to achieve a higher speed.

[12]Recall that in my implementation, the outputs from the motoneurons of 10 consecutive neural segments are averaged to control one mechanical joint (Subsection 2.5.2). Note that the parameters of the sinusoidal controller are set to match those from the output of the MNl at segment 50. Only the first five segments of controller 2 receive extra excitation.

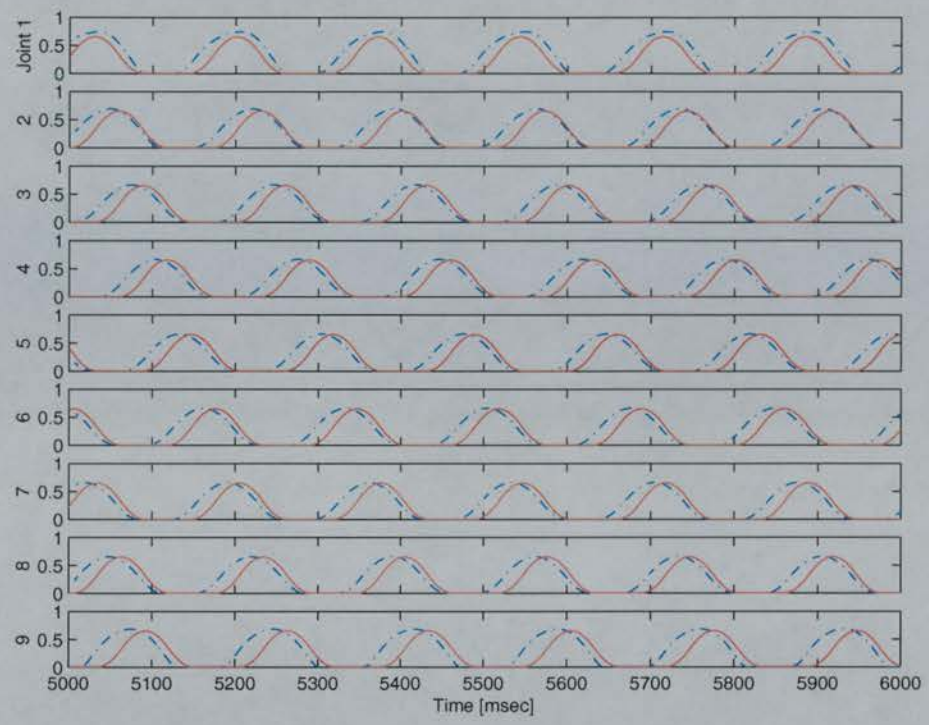


Figure 4.6: Averaged MNI activity from controller 2 with superimposed averaged MNI activity from a matched sinusoidal controller. The neural wave was generated with global excitation 0.9 and extra excitation 50%. Dashed lines represent outputs from controller 2. Solid lines represent outputs from the matched sinusoidal controller.

4.5.5 Intermediate discussion

Throughout the experiments, I used the change in maximum swimming speed as the criterion for robustness.[13] This is reasonable as the maximum speed an animal is able to achieve can affect its ability to survive. (Consider a predator chasing a large school of fish in a narrow passage, where turning becomes difficult.) Whenever Ijspeert and I evolve swimming controllers, we consider only the forward speed in the fitness function. Hence, measuring the maximum swimming speed implicitly measures the range of speeds which the model lamprey can achieve. (Recall that the fitness functions reward controllers which can increase their oscillation frequency - and hence the speed -

[13]Note that the maximum swimming speed was found after a comparison was made of all the achievable swimming speeds at different excitation combinations. Thus, I indirectly used the control inputs from the brainstem as an evaluation criterion.

as the global excitation from the brainstem increases.) In other words, I have indirectly studied the effect of scaling on the range of achievable swimming speeds. Other evaluation criteria such as the sharpness of turning, and the average speed vs. excitation combination could also be considered. In this dissertation, I have concentrated on forward swimming rather than turning abilities. Given that the simulation is deterministic, each swimming speed achieved at the corresponding excitation combination remains the same, and as a result, the average speed vs. excitation cannot be calculated. Since my model does not include metabolic rate, and also because there is no biological data on the lamprey for comparison, swimming endurance is not studied.

Comparison of the controllers' scaling performance with a matched sinusoidal controller shows that the latter's performance varies less than that of the CPG controllers over the range of scale factors considered. Thus, the matched sinusoidal controller is more robust. On the other hand, the CPG controllers generally perform better in terms of absolute maximum achievable speed. This suggests that the precise and uniform motoneuron pulse shapes and/or the variations in intersegmental phase lag and segmental output amplitude, make a difference to the performance of the complete system. Finally, the controllers presented here are deterministic. The robotic lamprey body in which they would be embedded is subject to both systematic and non-systematic variations. The study conducted here investigates the robustness of the controllers under consistent systematic variation of body parameters. It would be useful to find out how the controller performs under non-systematic variation in body parameters such as when the mass of each body link is varied randomly over time.

Based on their performance under scaling, the controllers tested fall into two groups: controller 2, which shows extremely good performance at the larger body scale (although not with increased body mass and muscle strength alone), and the rest of the controllers. All the controllers show performance variation with scaling. However, in all cases, the variation is smaller than the variation of body parameters. In Chapter 5 investigations are conducted to explain the distinctive scaling behavior of controller 2. Note that the evolved controllers used here were developed to maximize swimming performance at the nominal body scale. In Or et al. [2002] we attempted to use a Genetic Algorithm to evolve robust controllers using a fitness measure that

takes into account the variation in performance with body scaling. To summarize, we extended the fitness function described in Subsection 2.7.7 to allow evolutions of robust swimming controllers. In order to define robustness, a discrepancy function $discrepancy = \frac{|sp1.2-sp1|+|sp0.8-sp1|}{sp1}$ which takes the speeds at three different body scales (0.8x, 1.0x and 1.2x) was devised. A simple direct encoding scheme was used to encode the potential solutions. The controllers were developed in three evolutionary stages. In the first stage, segmental oscillators were evolved. In the second stage, the best two controllers evolved in previous stage were included in the initial populations of the new set of evolutions to produce the hybrid of these two oscillators. The best evolved controller, the *Hybrid Segmental Oscillator*, is capable of oscillating at a higher frequency than its parents. Furthermore, its motoneuron outputs have higher amplitude than those of its parents. (This is encouraging as it means that more muscle force can be recruited for high speed swimming.) In the final stage, the intersegmental couplings among 100 copies of any chosen segmental oscillator (such as the biological controller or the hybrid segmental oscillator) were evolved. We found that this was a less appropriate way to evolve controllers because the evolutions took about three months to complete and the evolved controllers were less responsive to brainstem modulations. (Note however that the best evolved CPG, the *Hybrid Robust Controller*, is more robust than the controllers investigated here under the discrepancy definition.) In Chapter 6, I show that evolving controllers based on swimming efficiency (which is another measure of performance) is a better approach.

4.6 Summary

This chapter has outlined the motivation for investigating firstly, the differences between the biological controller and controller 2 (Chapter 5), and secondly the evolutions of robust swimming controllers (in Or et al. [2002]). As a first step, I provided several definitions of robustness and discussed how they can be studied. From the discussion, I showed that the study of robustness in swimming speed against variation in body scale is important for both real and robotic fish. The effect of scaling body parameters was then discussed. Taking the biological controller and other evolved CPG

controllers, I compared their swimming performance across different body scales with that of an analytic sinusoidal controller. The results indicated that controller 2 behaves differently compared with the other controllers. The next chapter is devoted to explaining the unusual behavior of controller 2.

Chapter 5

Differences between the biological controller and controller 2

From the results of Experiment 4 presented in Chapter 4, I observed that controller 2 exhibits distinctive scaling behavior. Comparison of neural parameters (amplitude, frequency and phase lag) which correspond to the maximum speed at scales from 0.7 to 1.5x shows that the biological controller and controller 2 have different trends in terms of variation in the frequency of oscillation. To summarize, although the speed of the lamprey driven by either controller increases with the scale, the frequencies of oscillation (which correspond to the maximum speeds achieved by the controllers) vary in opposite directions. As the scale increases, the biological controller decreases its oscillation frequency. On the other hand, controller 2 increases its oscillation frequency with scale. Table 5.1 shows neural parameters and the corresponding speeds for these two controllers.

The remainder of this chapter is devoted to explaining the differences between the two controllers. Investigations at both behavioral and neural levels are carried out. At the behavioral level, the controllers are compared in terms of their swimming speed, leading to the discovery that the two controllers have different trends in varying the oscillation frequency. Controller 2 is also shown to have a larger overall phase lag. At the neural level, a comparison of the neural organization of the two CPGs is performed. Controller 2 is shown to have weaker overall connection strengths than the biological

Scale	Global	Extra [%]	Amplitude	Frequency [Hz]	Phase lag [%]	Speed [m/s]
0.7	0.7 **[0.9]**	75 **[25]**	0.69 **[0.74]**	5.32 **[5.49]**	0.93 **[1.07]**	0.399 **[0.423]**
0.8	0.6 **[0.9]**	150 **[25]**	0.64 **[0.74]**	5.18**[5.49]**	1.17 **[1.07]**	0.429 **[0.455]**
0.9	0.6 **[0.9]**	300 **[25]**	0.63 **[0.74]**	5.35 **[5.49]**	1.31 **[1.07]**	0.454 **[0.444]**
1.0	0.6 **[0.9]**	300 **[50]**	0.63 **[0.68]**	5.35 **[5.85]**	1.31 **[1.79]**	0.462**[0.467]**
1.1	0.5 **[0.9]**	175 **[50]**	0.58 **[0.68]**	4.98 **[5.85]**	1.49 **[1.79]**	0.464 **[0.514]**
1.2	0.5 **[0.9]**	175 **[50]**	0.58 **[0.68]**	4.98 **[5.85]**	1.49 **[1.79]**	0.473 **[0.544]**
1.3	0.5 **[0.9]**	175 **[75]**	0.58 **[0.66]**	4.98 **[5.95]**	1.49 **[1.85]**	0.475 **[0.550]**
1.4	0.4 **[0.9]**	275 **[100]**	0.51 **[0.65]**	4.50 **[5.99]**	1.69 **[1.95]**	0.474 **[0.557]**
1.5	0.4 **[0.9]**	200 **[100]**	0.51 **[0.65]**	4.52 **[5.99]**	1.72 **[1.95]**	0.486 **[0.553]**

Table 5.1: Neural parameters and speeds achieved by the biological controller and controller 2 across all scales under investigation. Parameters related to controller 2 are in brackets. *Global* and *Extra* represent the global and extra excitation respectively. The speed is the maximum achievable speed.

controller. Therefore, it can accept more global excitation to achieve higher oscillation frequencies for faster swimming at larger body scales.

Finally, through characterizing the lamprey model using a sinusoidal controller, I fill the gap between the trends in frequency variation and larger overall phase lags to solve the mystery of the different behavior of the two controllers.

5.1 An analysis of the anomalous speed curve of controller 2 in Experiment 4

A closer look at the speed curve of controller 2 in Figure 4.4D reveals that, to start with, the speed increases with the scale and then drops at scale 0.9x. After that, the speed increases with scale again. From Table 5.1, it is clear that the same global-extra excitation combination used at body scales 0.7x and 0.8x is used in scale 0.9x. In other words, the same neural wave (the same driving frequency) is applied to drive the lamprey with body scales from 0.7x to 0.9x. The drop in speed can be explained by using

a physical pendulum analogy (Subsection 5.1.1). At scale 1.0x, the excitation combination has changed (specifically, there is more extra excitation), and the swimming speed increases once again.

5.1.1 Using the forced pendulum as an analogy to understand the anomalous speed curve

In order to understand the anomaly of the speed vs. scale curve, a simple forced physical pendulum can be considered to be analogous to a mechanical lamprey segment.

The four factors which affect the behavior of a physical pendulum are: mass, moment of inertia, damping and external driving force. In the case of a lamprey segment, the water force is the damping factor while the torque generated by the motoneurons is the periodic driving force. For the pendulum, air is the damping factor while the external driving force sets the pendulum into motion. Since the four factors governing the physical pendulum system are analogous to those controlling the lamprey system, they are at some level similar. Thus, the pendulum can be used as a simple model for the lamprey. In the following explanation, let's consider a lamprey body of two segments. One is fixed while the other is free to oscillate between the left and the right. The pivot is at the joint where the two body segments connect.

The angular frequency vs. amplitude curve for a physical pendulum is bell-shaped [Serway et al., 2000]. The peak appears at a point where the frequency of the external driving force matches the natural frequency (ω_0) of the pendulum. We call this frequency the *resonance*. To achieve maximum swinging amplitude, I vary the driving frequency to match the natural frequency of the pendulum. The scale vs. amplitude curve for a 2-segment lamprey is also similar to a bell-shape. To achieve maximum amplitude, we vary the scale (which indirectly varies the physical properties of the lamprey body), changing the natural frequency of the lamprey body so that it matches the frequency of the external driving force (the torque produced by the muscles). Note that since the neural wave remains unchanged, the frequency of the driving force remains the same (recall from Section 5.1 that the same neural wave is used to drive the body at scales from 0.7x to 0.9x). At resonance, the natural frequency of the lamprey body matches the oscillation frequency of the neural wave. Energy is then transferred

to the lamprey body under the most favorable condition (i.e. the external driving force and the velocity of a lamprey segment are in-phase). This leads to maximum swinging amplitude.

The scale vs. speed curve is similar to the scale vs. amplitude or the frequency vs. amplitude curve. Suppose that initially the frequency of the driving force matches the natural frequency of the body. As we increase the scale, the natural frequency decreases. Since the neural wave (and therefore the driving muscle force) remains the same, the amplitude decreases.[1] This in turn reduces the water force, as it is proportional to the amplitude. Given that the water force is also proportional to the square of speed, the speed decreases too.

Note that the dip in controller 2's speed curve at scale 0.9x is caused by the fact that at this scale, the natural frequency of the body does not match the frequency of the driving force.

5.2 Repetition of Experiment 4 with the new speed calculation algorithm and higher resolutions in the excitation grids

As mentioned in Section 3.5, the model lamprey can turn in any direction as I vary some of the parameters such as body scales. Hence, calculating the swimming speeds using the simple speed calculation method (i.e. through the direct distance traveled by the head) may produce less accurate results. Also, the resolutions of the excitation grids used in previous experiments might have been too low. Higher achievable maximum speeds might have been missed for controllers whose swimming speed is very sensitive to changes in excitation.

Using the new speed calculation algorithm described in Section 3.5 with higher resolutions in the excitation grids (the step between neighboring extra excitations is reduced from 25% to 10%), I repeated Experiment 4 with the three CPG controllers.

[1]The damping factor for a pendulum is $-b \cdot |\dot{x}|$ while that for the lamprey is $-b \cdot |\dot{x}|^2$. Therefore, the amplitude of motion is not proportional to the size of the driving force. Note that b is the damping coefficient and $\dot{x}$ is the velocity.

The results are shown in Figure 5.1. Once again, both the biological controller and controller 3 behave similarly. Controller 2 still performs much better at the larger body scales. With the reduction in interval between each extra excitation pair, the dip at scale 0.9x disappears. Table 5.2 shows the excitation combination as well as the parameters for the neural signals (from controller 2) which correspond to maximum speed at each scale. Again, the frequency of oscillation increases with the scaling factor. At the normal body scale, the biological controller and controller 3 achieve the same maximum speed as before. However, the speed of controller 2 is higher in this experiment. (Compare Table 4.2 with Table 5.3.) There are two reasons for this. Firstly, in the original experiment, the excitation combination which corresponds to the maximum speed of 0.467 m/s is (0.9, 50%). In the repeated experiment, the maximum speed of 0.49 m/s is achieved under (0.9, 40%). Thus, the lower resolution of the original experiment missed the higher speed found in the repeated experiment. Secondly, the new speed calculation algorithm takes turning into account. With excitation combination (0.9, 40%), the speed computed using the new algorithm is 0.49 m/s which is slightly higher than the one obtained with the original speed calculation method. (The angular speed is 0.24 degrees/sec.)

Scale	Global	Extra [%]	Amplitude	Frequency [Hz]	Phase lag [%]	Speed [m/s]
0.7	0.9	20	0.75	5.41	0.92	0.410
0.8	0.9	30	0.73	5.59	1.26	0.450
0.9	0.9	30	0.73	5.59	1.26	0.481
1.0	0.9	40	0.70	5.78	1.56	0.490
1.1	0.9	40	0.70	5.78	1.56	0.529
1.2	0.9	50	0.68	5.85	1.70	0.544
1.3	0.9	60	0.67	5.88	1.79	0.553
1.4	0.9	100	0.65	5.99	1.95	0.557
1.5	0.9	120	0.65	6.02	1.96	0.556

Table 5.2: Neural parameters and speed achieved by controller 2 across all scales for the repeated Experiment 4. *Global* and *Extra* represent the global and extra excitation respectively. The speed is the maximum achievable speed.

When Table 5.4 is compared with Table 4.6, it can be seen that the range of percentage change in speed for controller 2 is slightly larger in the new experiment. On the other hand, the other two controllers have a smaller range in terms of percentage

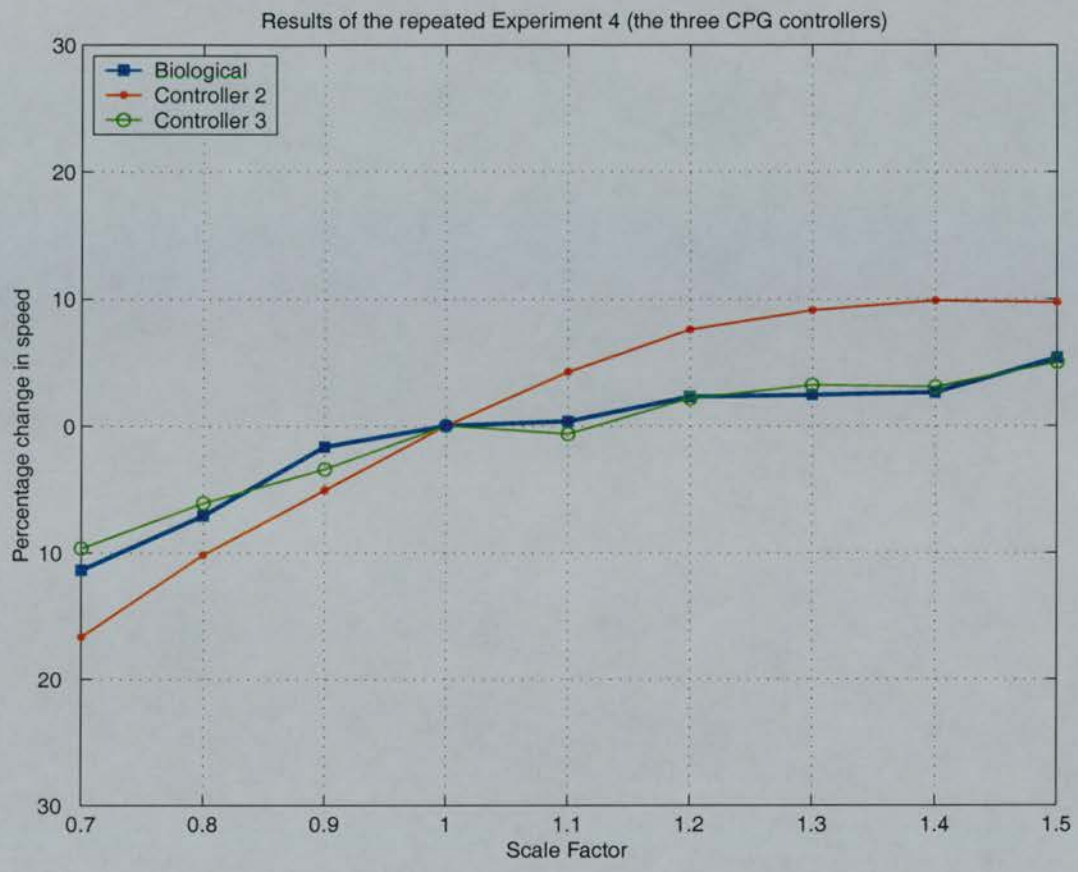


Figure 5.1: Performance of the CPG controllers in the repeated Experiment 4 using the new speed calculation algorithm and higher excitation resolutions

change in speed.

5.3 A comparison of the neural organization of the biological controller and controller 2

Comparing the controllers in terms of neural organization is very difficult, partly because different types of neurons are interconnected to provide an emergent behavior for the CPG, and also because of the extra excitations given to neurons at the head. However, it is still better to compare the connection strengths at the head of the two CPGs. The reason for this is that since the head segments receive extra excitations,

Controller	Speed [m/s]
Biological	0.462
Controller 2	0.490
Controller 3	0.430

Table 5.3: Speeds achieved by the different controllers with their original configurations in the repeated Experiment 4

Controller	Speed [m/s] (0.7x)	Speed [m/s] (1.5x)	Difference in speed [%]
Biological	0.409	0.487	-11.4, 5.34
Controller 2	0.410	0.556	-16.4, 13.4
Controller 3	0.388	0.451	-9.65, 4.99

Table 5.4: Speeds achieved by the different controllers with different body scales in the repeated Experiment 4

any difference between the two controllers should be amplified for easier comparison. Hence, a cross comparison between the controllers based on this criterion may lead to a better understanding of why the two controllers behave differently (as shown in Table 5.1). Using the connection weight matrix for the two controllers from Table 4.1 and the method for computing the calibrated connection strengths under post-synaptic implementation (Subsection 3.1.1), the calibrated connection strength matrix for the two controllers is shown in Table 5.5. The corresponding Hinton diagrams are provided in Figure 5.2 for visualization purposes.

Table 5.5 and Figure 5.2 show that the overall calibrated connection strengths (both excitatory and inhibitory) for controller 2 at the head segments are weaker than the corresponding ones for the biological controller. Hence, compared with the biological controller, controller 2 is able to accept more global excitations from the brainstem. The largest global excitation it can receive while being able to oscillate is 0.9. To achieve the maximum speed at increasing body scale, it stays at 0.9 to maintain the

From: / *To:*	EINl	LINl	CINl	CINr	BS
MNl	1.2 **[0.2]**	-	-	-2.0 **[-0.17]**	5.0 **[5.0]**
	0.09 **[0.17]**	-	-	-0.18 **[-0.09]**	
EINl	0.4 **[0.07]**	-	-	-2.0 **[-1.0]**	2.0 **[2.0]**
	0.08 **[0.04]**	-	-	-0.17 **[-1.0]**	
LINl	13.0 **[2.6]**	-	-	-1.0 **[-0.1]**	5.0 **[5.0]**
	1.18 **[2.17]**	-	-	-0.08 **[-0.07]**	
CINl	3.0 **[0.25]**	-1.0 **[-0.25]**	-	-2.0 **[-2.0]**	7.0 **[7.0]**
	0.6 **[0.21]**	-0.09 **[-0.08]**	-	-0.17 **[-0.33]**	

Table 5.5: Calibrated connection strength matrix for the biological controller and controller 2. Each matrix element has two rows. Numbers in the first row represent connections at the head while numbers in the second row represent connections in the middle of the CPG. Numbers in bold represent the calibrated connection strengths of controller 2. Excitatory and inhibitory connections are represented by positive and negative signs respectively. Left and right neurons are indicated by *l* and *r*. *BS* stands for brainstem. For each type of neuron, the connection weight from the *BS* is the same regardless of the location in the swimming CPG. Note that the magnitude of the calibrated connection strengths in controller 2 are generally lower than the corresponding ones in the biological controller.

highest possible frequency (refer to Table 5.1). In order to keep up with more demand in frequency and phase lag, more extra excitations are required.[2] As for the biological controller, since its overall calibrated connection strengths are stronger, its network fails to oscillate earlier than controller 2's. Hence, it is unable to receive more global excitation, and instead it has to lower the global excitations. As a result, the oscillation frequency decreases. The only way to increase frequency and phase lag any further is to increase the amount of extra excitation.

[2]For an explanation as to why the controller needs to increase frequency and phase lag at larger body scales, refer to Subsection 5.4.4.

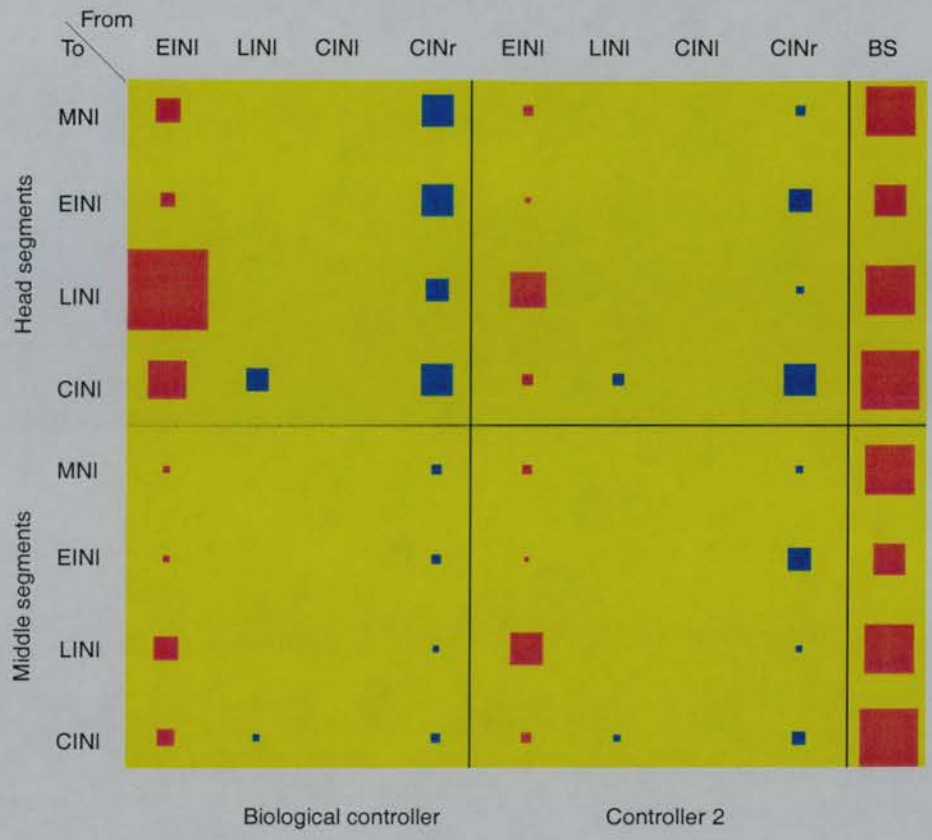


Figure 5.2: Hinton diagrams for the calibrated connection strength of the biological controller and controller 2. Boxes in red represent excitatory connections while boxes in blue represent inhibitory connections. The size of the box is proportional to the magnitude of the connection weight which the box represents. Note that only connections from the EINI neurons and from the brainstem (BS) are excitatory.

5.4 Characterization of the mechanical model using a sinusoidal controller

In the previous section, I found that there are differences in neural organization between the biological controller and controller 2. In order to understand the difference in behavior shown by the two controllers, it is necessary to understand how the mechanical lamprey model behaves under different parameters of control signals. In this section, I explore the behavior of the mechanical lamprey model through the use of

a sinusoidal controller.[3] The reason for choosing this controller is that all aspects of the control signals (such as amplitude, frequency and phase lag) can be varied independently. Hence, the effect of each control variable on the swimming speed can be studied in detail.

Before analyzing how the scale and control signals affect the swimming speed, mechanical simulations with different body scales and controller variables are performed. As in Experiment 4 in the study of robust controllers, the physical parameters and forces are scaled in all three dimensions (refer to Section 4.3 for details). Since turning is unavoidable, the new speed calculation algorithm developed in Section 3.5 is used.

The ranges under investigation are shown as follows:

- Amplitude: 0.4 : 0.025 : 1.0
- Frequency: 2.0 : 0.25 : 15.0 [Hz]
- Phase lag: 0.6 : 0.05 : 2.0 [%]
- Scale: 0.8 : 0.1 : 1.5

5.4.1 The effect of oscillation frequency and scale on swimming speed

As I am interested in the effect of scaling on swimming speed, I plotted the speed(scale, frequency) surfaces with different amplitudes and phase lags. By comparing the surfaces, the effect of the latter two variables on the speed(scale, frequency) relation can be observed. In the discussion which follows, I focus on the speed surface with amplitude fixed at 1.0 and constant phase lag 2.0%. The reason for this is that at very large amplitude and phase lag, the effect of oscillation frequency and scale on the swimming speed is the most profound (Figure 5.3).

At any scale, there are two resonance: one at lower frequency and the other at a much higher frequency (hereafter called $resonance_1$ and $resonance_2$ respectively). As the oscillation frequency increases, the speed increases and then drops. This is due to the resonance effect (mentioned in Subsection 5.1.1).

[3]For details of the sinusoidal controller, refer to Subsection 4.5.4.

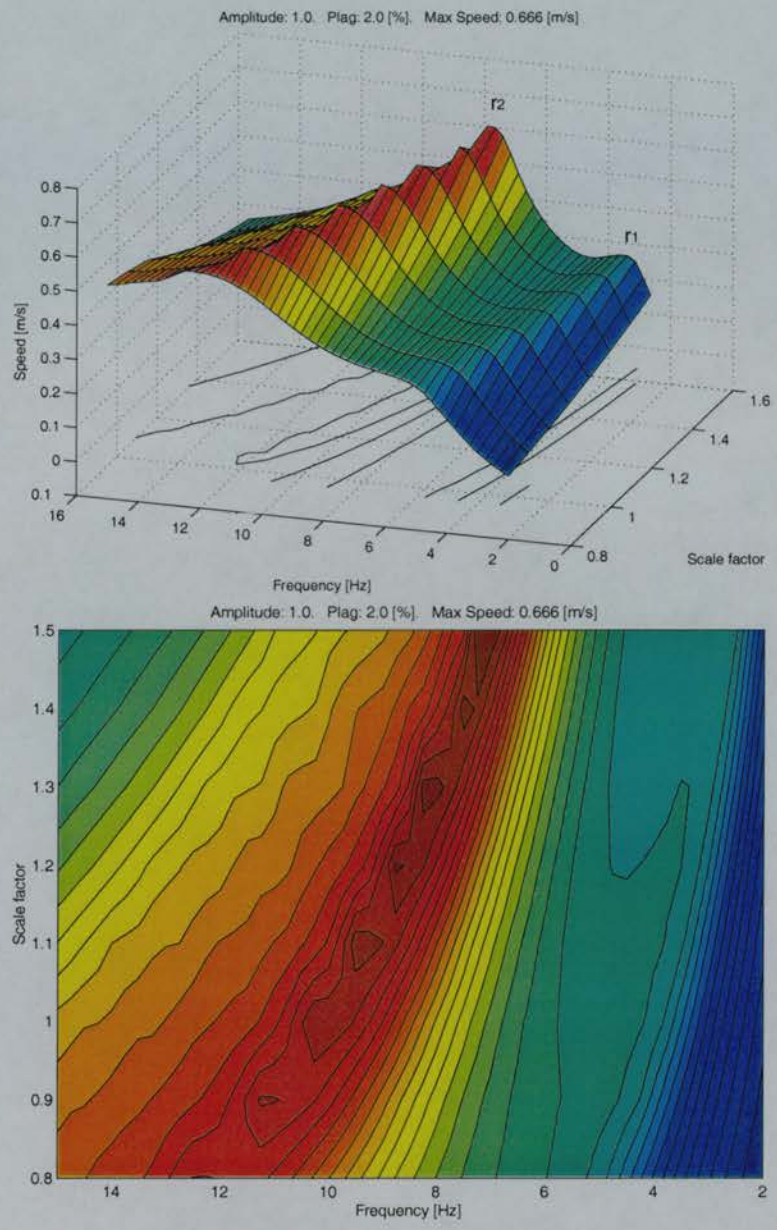


Figure 5.3: Effect of the oscillation frequency and scale on the swimming speed. Amplitude and phase lag are fixed at 1.0 and 2.0% respectively. Surface plot (*top*) and contour plot (*bottom*). r_1 represents $resonance_1$ while r_2 represents $resonance_2$.

Across all scales, the maximum achievable speed at $resonance_2$ is always higher than that at $resonance_1$ (a possible explanation follows). As the scale increases, the maximum achievable speed at both resonances increases. Meanwhile, both resonances shift to lower frequencies since, at larger body scales, the natural frequencies of the lamprey body decrease due to higher inertia.[4]

5.4.2 Possible reasons for the multiple resonance

The mechanical lamprey model consists of 10 links. The first four links are larger and heavier, while the middle links are slightly smaller and lighter, and the links towards the tail are very small and light. Thus, there is a big difference in physical characteristics between the very front and the very end of the body. Hence, the entire lamprey body should possibly be modeled by two or three groups of pendulums, each of which has its own natural frequency. The heavier segments at the head require low frequency to achieve resonance ($resonance_1$) while the lighter tail segments require a much larger frequency for resonance ($resonance_2$) to take effect. This may explain why there are two (or even three) resonances in some of the speed surfaces. To test this hypothesis, I used a sinusoidal controller with amplitude and phase lag fixed at 1.0 and 2.0% respectively,[5] while the frequency varies from 1 to 15 Hz in steps of 0.2 Hz. Since a body link is supposed to have a larger maximum deflection angle when its natural frequency matches the frequency of the driving force, I recorded the turning angle at each body joint during the last second of each mechanical simulation. The corresponding deflection angle (the maximum amount of turning) at each body joint is calculated. The results are shown in Figure 5.4.

The figure shows that at the lower end of the frequency range (about 1.5 Hz), the head segments turn more than those in the middle of the body because the low driving frequency matches the resonance frequency of the heavier head segments. For frequency at about 5 Hz, the body links in the middle of the body turn more than the head because they are lighter and therefore have higher natural frequencies. As the driving frequency continues to increases, the links towards the tail turn the most,

[4] Although the muscle force is scaled as well, it is scaled as the third power rather than the fifth power as for inertia.

[5] So as to match the description given in Subsection 5.4.1.

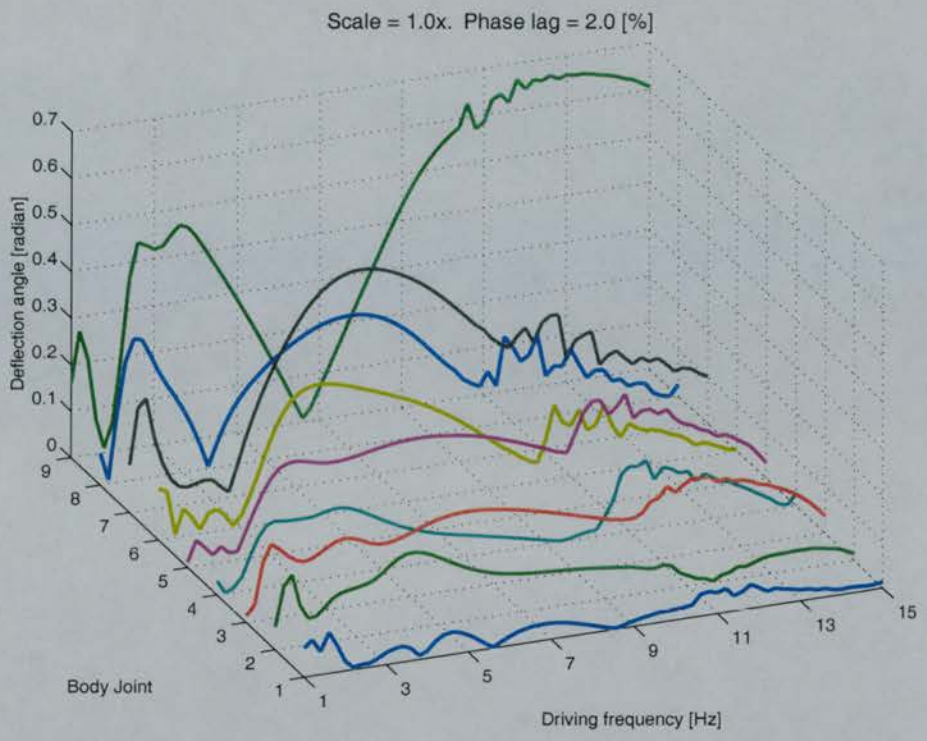


Figure 5.4: Maximum deflection angle at each body joint under different driving frequencies.

because they have the highest natural frequencies. It is interesting to note that the tail segments also turn more at the lower end of the frequency range. This implies that they have more than one natural frequency. Note that as the links are inter-connected, resonances at the head segments may have an effect on the tail segments. This effect is facilitated by the assumption that there is no parallel water force at the tail segments.

From [Bainbridge, 1958], the tail beat frequency is an important factor in determining the swimming speed. For a fixed body scale, as the speed increases with frequency, $resonance_2$ (at the tail) has a larger effect on the speed than $resonance_1$ (at the head). This explains why the speed region near $resonance_2$ has a higher overall speed than that of $resonance_1$.

To investigate the scale effect on different body parts, I repeated this experiment with body scales at 0.8x, 1.2x and 1.4x of the original. The results are shown in Figure 5.5.

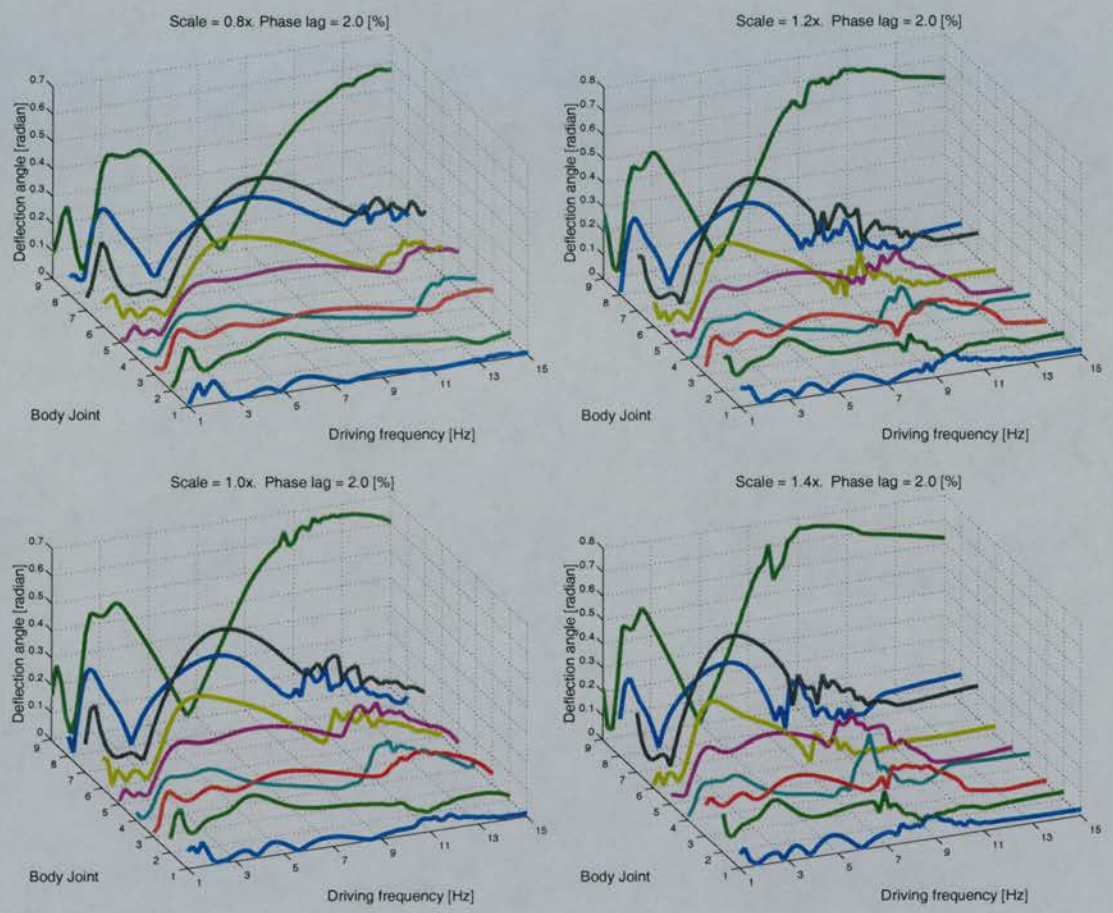


Figure 5.5: Maximum deflection angle at each body joint under different driving frequencies and body scales. The scale increases from 0.8x to 1.4x in steps of 20%. Amplitude is fixed to 1.

This figure shows that as the scale increases, the natural frequency of the three body partitions decreases. Thus, the peaks appear at lower frequencies.

5.4.3 The effect of increasing the amplitude on the speed(scale, frequency) relation

When the amplitude of the control signal is increased (while phase lag is kept constant), the maximum achievable speed increases (Figure 5.6). This is because larger amplitude is able to produce higher torques to overcome the water inertia.

5.4.4 The effect of increasing the phase lag on the speed(scale, frequency) relation

When the phase lag of the control signal is increased (while amplitude is kept constant), the maximum achievable speed of each surface increases and is reached at a higher frequency. It is not clear why this is the case. A possible explanation is as follows: Recall that I use the average motoneuron output of 10 neural segments to control one mechanical joint (Subsection 2.5.2). Increasing the phase lag while keeping the amplitude constant implies that the pulse width of the averaged motoneuron output becomes wider. From [Grillner and Kashin, 1976, Wallén and Willams, 1984] and experimental data (see below), speed is inversely proportional to both burst duration (pulse width) and cycle duration. Hence, in order to keep up with the maximum speed, the oscillation frequency has to be increased. Since the higher the oscillation frequency, the faster the lamprey swims [Grillner et al., 1995], the speed increases at a higher frequency. For example, in Figure 5.8, the maximum speed as indicated by point B shifts to a higher resonance frequency as the phase lag increases. This seems to be why as the phase lag increases (when the amplitude is fixed), the maximum speed increases and appears at a higher frequency.

In order to verify that the speed of the model lamprey decreases as the pulse width increases (due to an increase in phase lag and the way I use the motoneuron outputs to control the body joints), I conducted the following experiment: For simplicity, I keep the neural characteristics (amplitude, frequency and phase lag) constant at 1.0, 6.0 Hz and 1% respectively. Instead of using the averaged motoneuron outputs from 10 neural segments to control one body joint, I use the average of the motoneurons outputs from an increasing number of segments (from 1 to 20). Averaging the motoneuron outputs

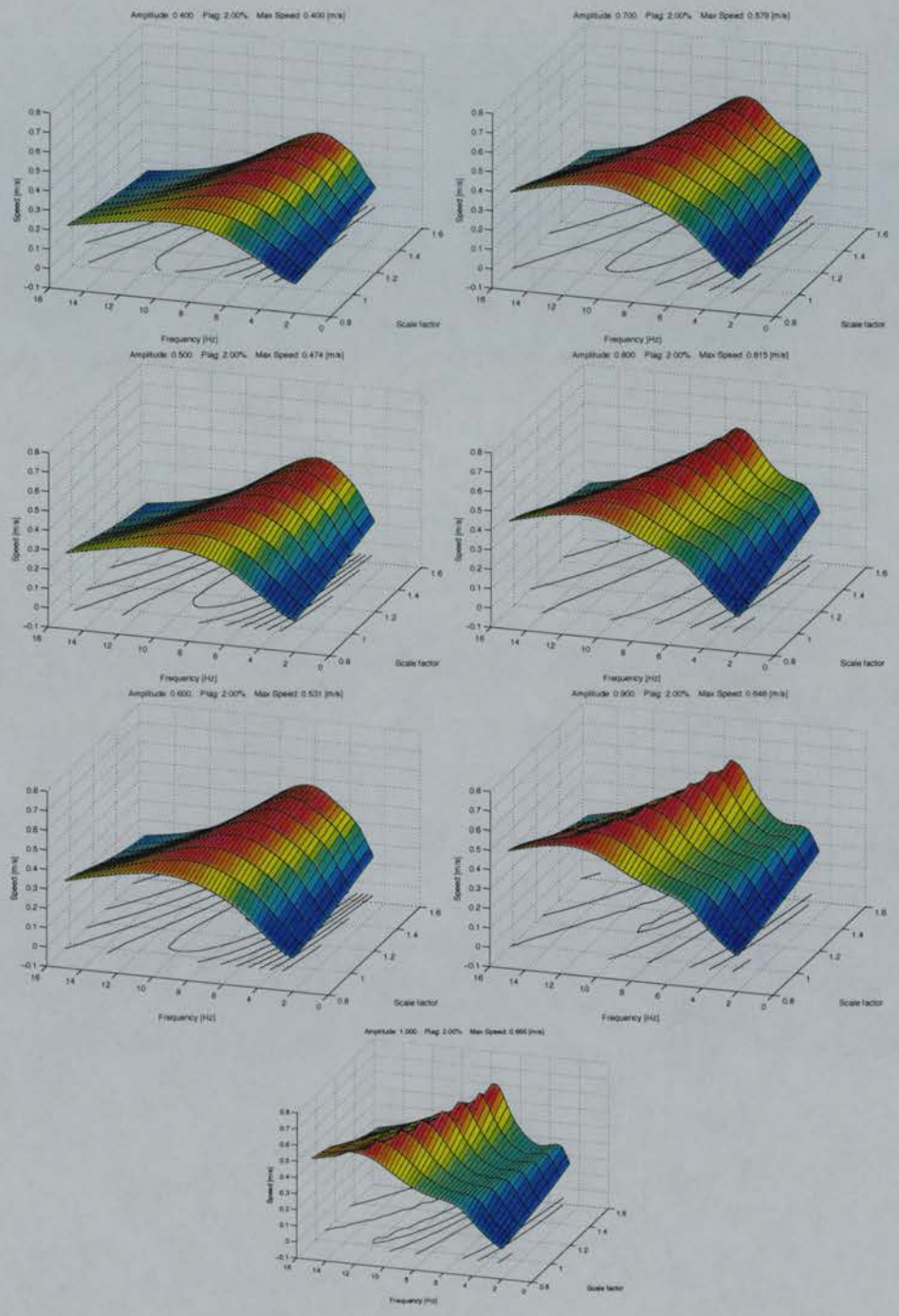


Figure 5.6: Effect of increasing the amplitude on the speed(frequency, scale) relation (phase lag is fixed at 2%).

from less than 10 segments should reduce the pulse width. Conversely, averaging the motoneuron output from more than 10 segments should increase the pulse width. The results are shown in Figure 5.7.

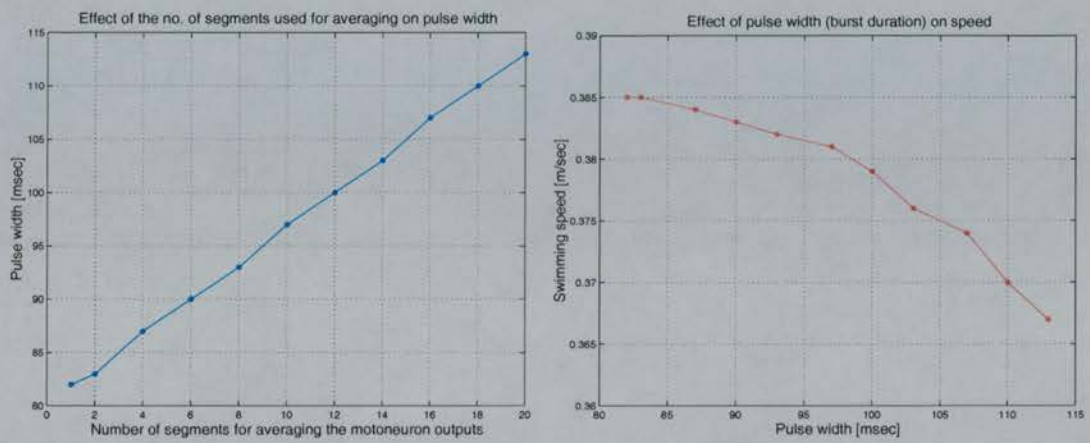


Figure 5.7: Effect of burst duration on swimming speed. The effect of the number of segments used to control a body joint on pulse width (*left*). The effect of pulse width on swimming speed (*right*).

Figure 5.7 shows that increasing the number of neural segments to control a body joint (while keeping the other neural parameters constant) increases the pulse width. This corresponds to a reduction in swimming speed. Thus, the model lamprey is shown to behave like the real fish in terms of having its speed inversely proportional to burst duration.

5.4.5 Characterization of the mechanical model at nominal scale

In order to understand further the mechanical model across a wide range of amplitudes, frequencies and phase lags, an extra set of experiments is performed. As I am only interested in the general tendency, the variables are sampled at a coarse scale as follows:

- Amplitude: 0.4 : 0.05 : 1.0
- Frequency: 15 : 0.50 : 25 [Hz]

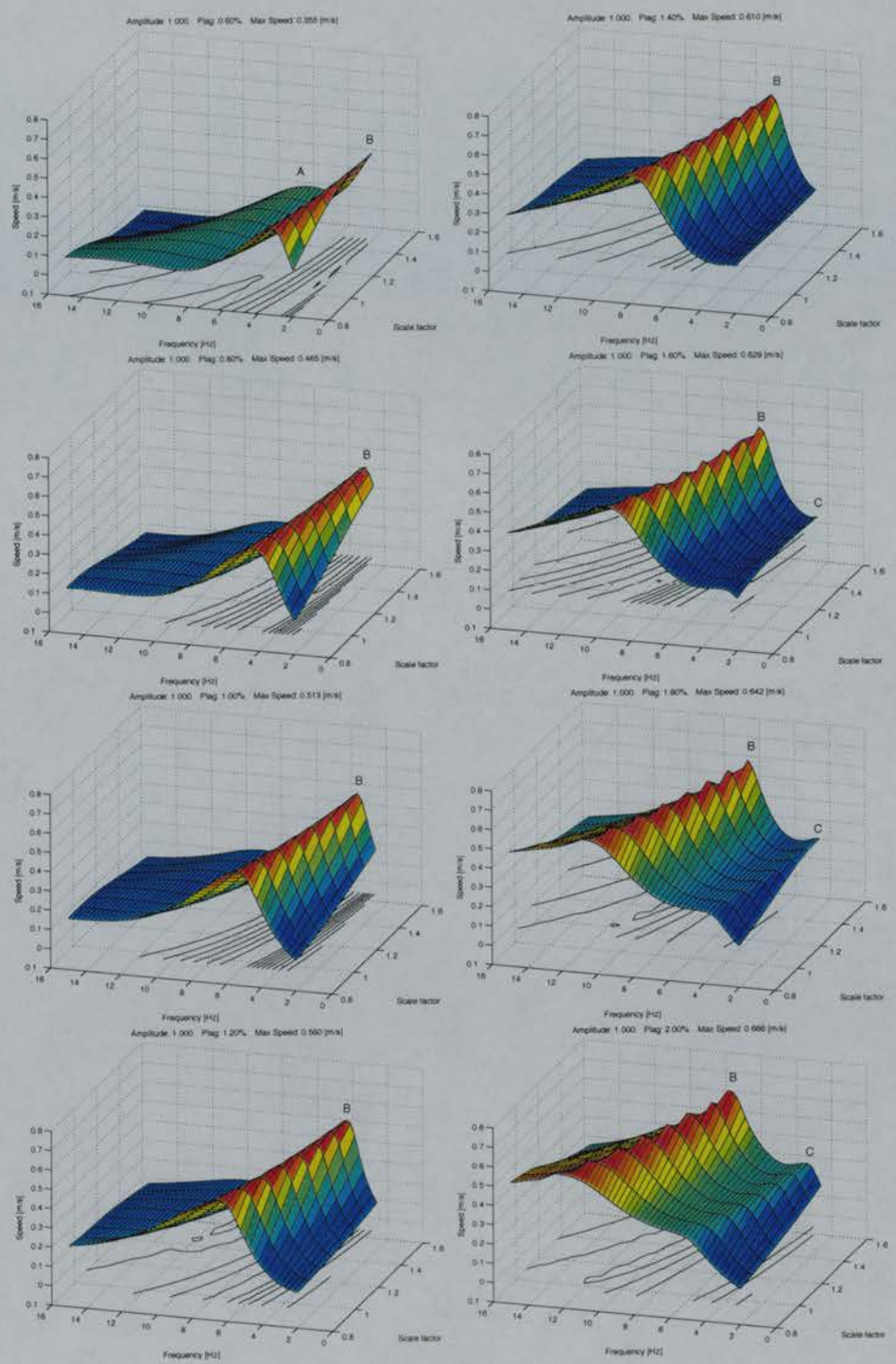


Figure 5.8: Effect of increasing the phase lag on the speed(frequency, scale) relation (amplitude is fixed at 1.0). A, B and C are peaks corresponding to maximum speed (refer to the text for explanation).

- Phase lag: 0.0 : 0.2 : 3.6 [%]
- Scale: 1

I combined the fine and coarse surfaces (Figure 5.9) and observed that at fixed amplitude of 0.8 and scale at 1.0x, there are two resonances (within the 2 to 25 Hz range) at each phase lag within the range under investigation, one at about 2.5 Hz and the other at about 10 Hz. The former corresponds to a much higher swimming speed than the latter. It shifts to about 9 Hz as the phase lag increases. At the same time, the swimming speed increases accordingly. Note that the speed surface made from the coarse data has negative speed (-0.12 m/s) at no phase lag. This is due to the effect of the lamprey wriggling.

The maximum speed obtained is about 0.6 m/s at frequency 9.5 Hz and phase lag 2.0%. This is similar to that obtained in Ijspeert [1998]. In his case, the maximum speed and phase lag are the same as mine, but the frequency he uses is set at 10 Hz.

Plotting the speed surface with phase lag from 0.0 to 3.6% and frequency from 15 to 25 Hz reveals that the speed does decrease as phase lag and frequency continue to increase (Figure 5.10).

According to Ijspeert [1998], this is because "*the elasticity and the damping of the muscles prevent them making large contractions at higher frequencies*". As the model lamprey cannot oscillate at such a high range of frequencies, the swimming speed decreases. Within the forced pendulum analogy, this corresponds to having a frequency far away from any of the resonance frequencies. That is, being in the higher end tail of the bell-shaped curve (personal communication with Ijspeert). The reason why the speed curve is bell-shaped across the phase lag range can be explained as follows: At phase lag of 1%, there is a 100% lag between the head and the tail. Thus, the body forms a S-shape. The lateral forces on each side of the body cancel out and the lamprey propels forward. If the phase lag is reduced to 0.5%, the head and the tail point to the same direction to form a C-shape, which results in the lamprey turning. The phase lag has to be over 0.5% for the ends of the lamprey to point in opposite directions. As the phase lag continues to increase, more body links are required to propagate the neural wave. Given that the mechanical model has only 10 body links, this cannot be accomplished and therefore the swimming speed decreases.

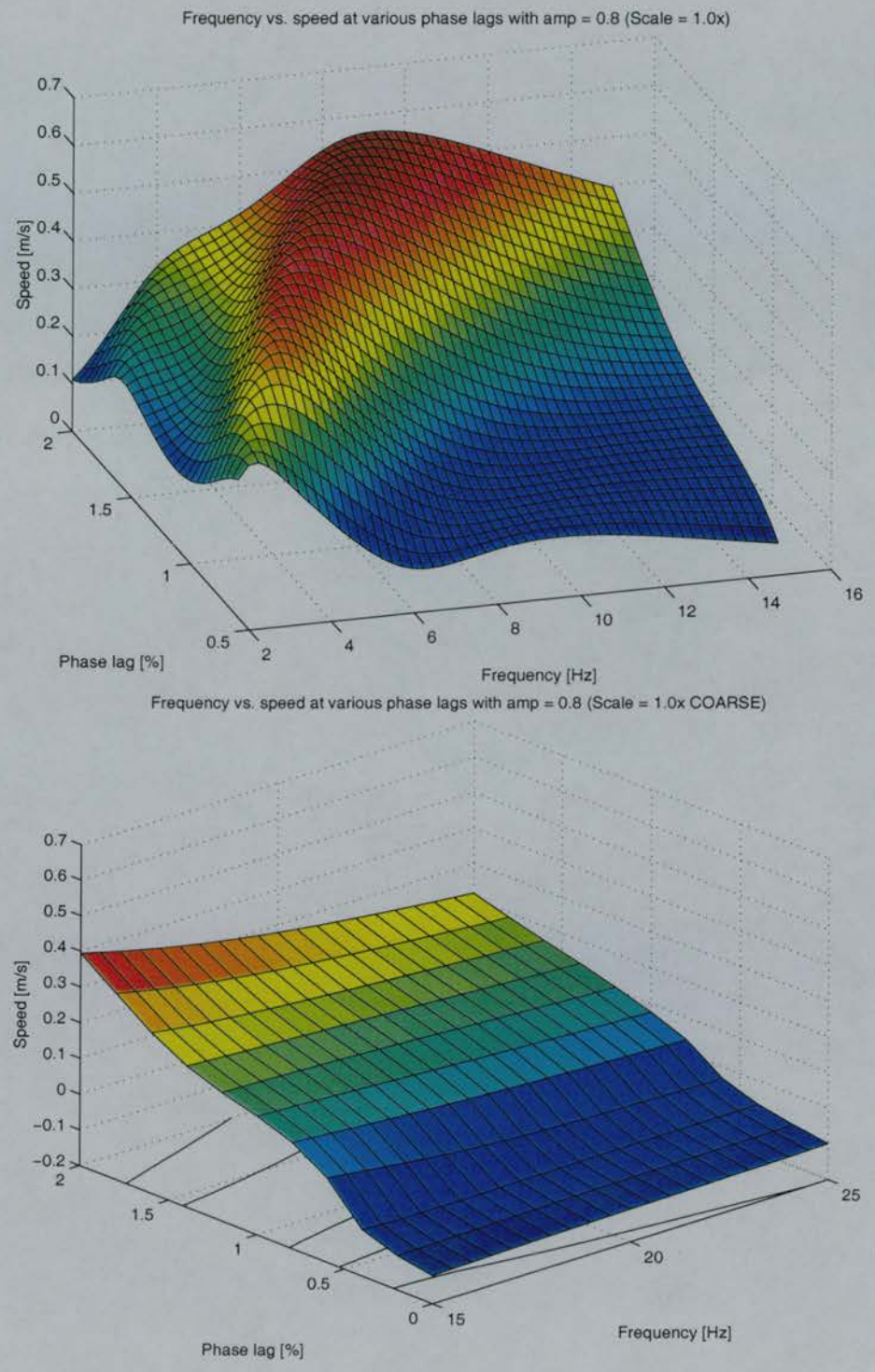


Figure 5.9: Speed(frequency, phase lag) surface with fine (*top*) and coarse sampling (*bottom*).

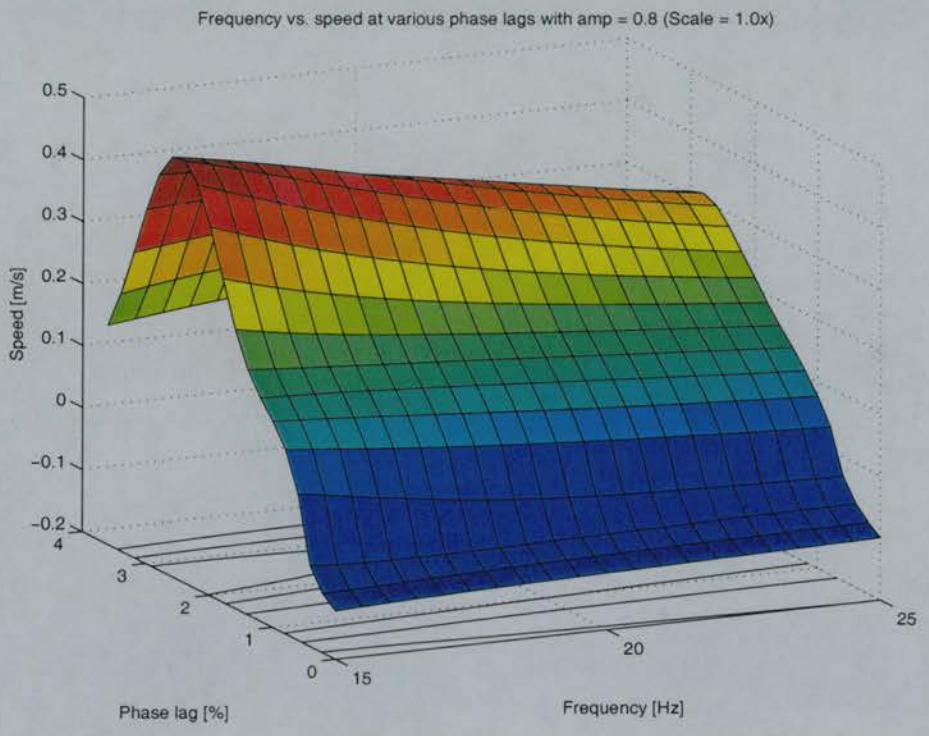


Figure 5.10: Swimming speed at higher frequency and phase lag (coarse sampling).

5.5 Explanations for the differences between the biological controller and controller 2

This section finalizes the investigation of the differences between the biological controller and controller 2. To recapitulate, the biological controller and controller 2 behave very differently. Although the maximum speed achieved by both controllers increases with scale, the oscillation frequencies corresponding to these maximum speeds change in opposite directions. In other words, as the scale increases, the frequency of oscillation of the biological controller decreases, while the frequency of oscillation of controller 2 increases (Table 5.1). The behavioral difference of the two controllers can be explained as follows.

The mechanical lamprey model used for this thesis exhibits multiple resonances across the frequency range and scales under investigation (Figure 5.11).

As the scale increases, the natural frequencies of the lamprey body decrease. The biological controller reduces its oscillation frequency to match the lower natural frequency of the body (path P2). On the other hand, due to its unique neural organization, controller 2 is able to accept more global excitations from the brainstem (before failing to oscillate) to increase its oscillation frequency to match the higher natural frequency (path P1). In other words, the controllers are aiming for two different natural resonances.

Since higher frequency leads to higher swimming speed, the model driven by controller 2 is able to achieve a higher swimming speed than when it is driven by the biological controller. Given that I chose the neural wave which corresponds to the maximum swimming speed as the criterion in the study of robust controllers, the swimming speed which corresponds to the neural wave with higher oscillation frequency was chosen at each scale. That is why controller 2 did not reduce its oscillation frequency to match the lower natural frequency as the biological controller does. As for the reason why controller 2 does not oscillate at the higher natural frequency (around 11 Hz at 0.8x and around 7 Hz at 1.5x), this can be explained by the fact that the maximum frequency at which it can oscillate is about 6 Hz. Similarly, the reason that the biological controller does not aim for the 10 Hz oscillation is that it is not capable of oscillating

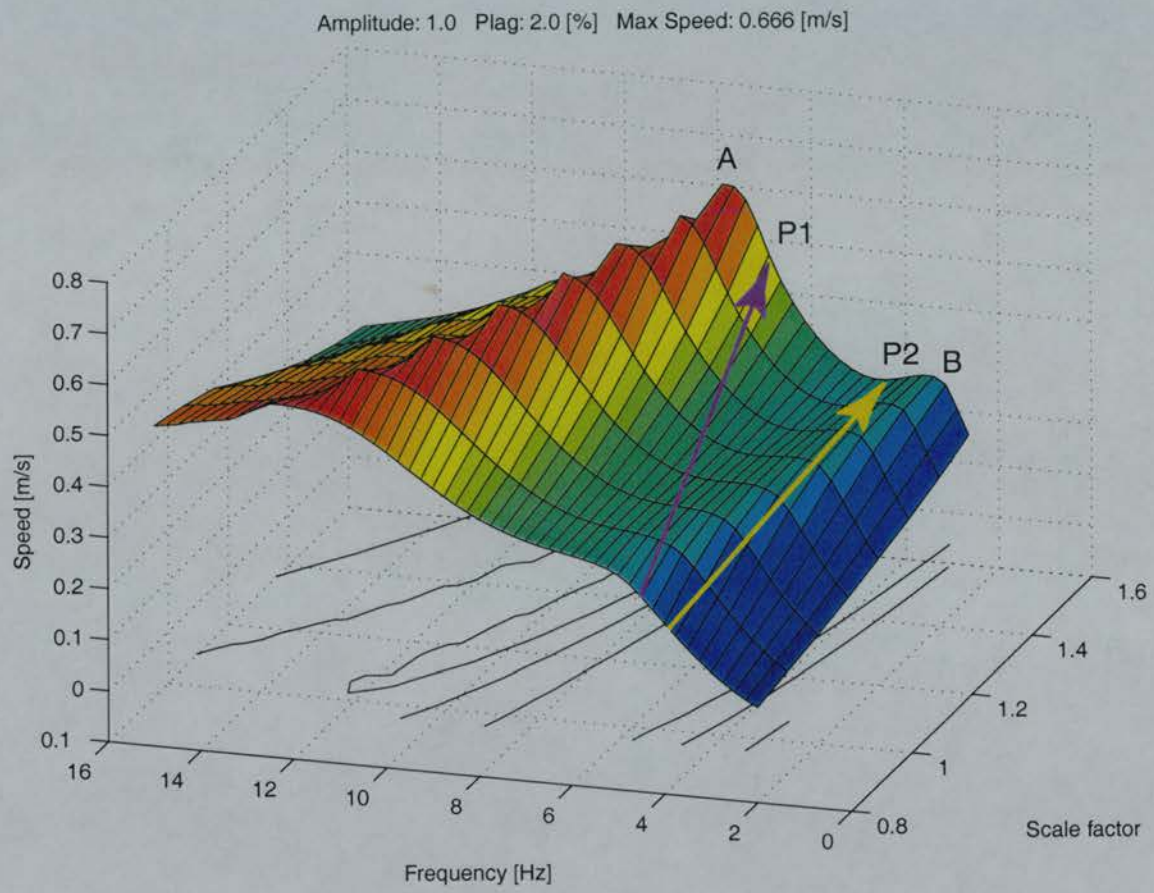


Figure 5.11: Effect of the oscillation frequency and the scale on the swimming speed. Amplitude is fixed at 1.0 and phase lag is fixed at 2%. A and B stand for the peaks corresponding to the maximum speed achieved at the higher and lower resonance respectively. P1 and P2 stand for the path followed by controller 2 and the biological controller respectively.

at such a high frequency.

5.6 Summary

This chapter was devoted to explaining the unusual speed curve of controller 2. First, a forced pendulum was used as an analogy to understand the anomalous dip in the speed curve at scale 0.9x. Then, Experiment 4 mentioned in Chapter 4 was repeated using the new speed calculation algorithm with higher excitation combination grids. The speed curves of the controllers in the repeated experiment show the same trends as before. However, the dip at scale 0.9x of controller 2's speed curve disappears when the resolution of the extra excitation grid is increased. A comparison of the neural configuration for the biological controller and controller 2 was performed. Using results from neural analysis, controller 2 was shown to be capable of producing higher overall frequencies and phase lags while being able to withstand more excitation before it fails to oscillate. Finally, through characterizing the mechanical lamprey model with a sinusoidal controller, I discovered that the lamprey body exhibits multiple resonances from the speed(scale, frequency) surface. This leads to the explanation that the biological controller and controller 2 were targeting different resonances (and therefore have different trends in frequency of oscillation).

Finally, there are a few points that are worth mentioning in regard to this chapter.

- Ijspeert and I use different methods of calculating the swimming speed. Ijspeert did not take turning into consideration (personal communication), whereas my method calculates the forward speed while taking turning into account. Although Ijspeert and I use the positions of the head to calculate the swimming speed, the two time instances which I considered correspond to the two instances at which the lamprey body are in-phase. Hence, my speed calculation method is more accurate (refer to Section 3.5 for details). For interested readers, an illustration of a lamprey during turning is shown in Figure 5.12. For an illustration of the normal straight forward swimming gait, refer to Figure 3.7.

- Although there are slight differences in the implementation of speed calculations and the use of segmental signals to control the torque, a comparison of my

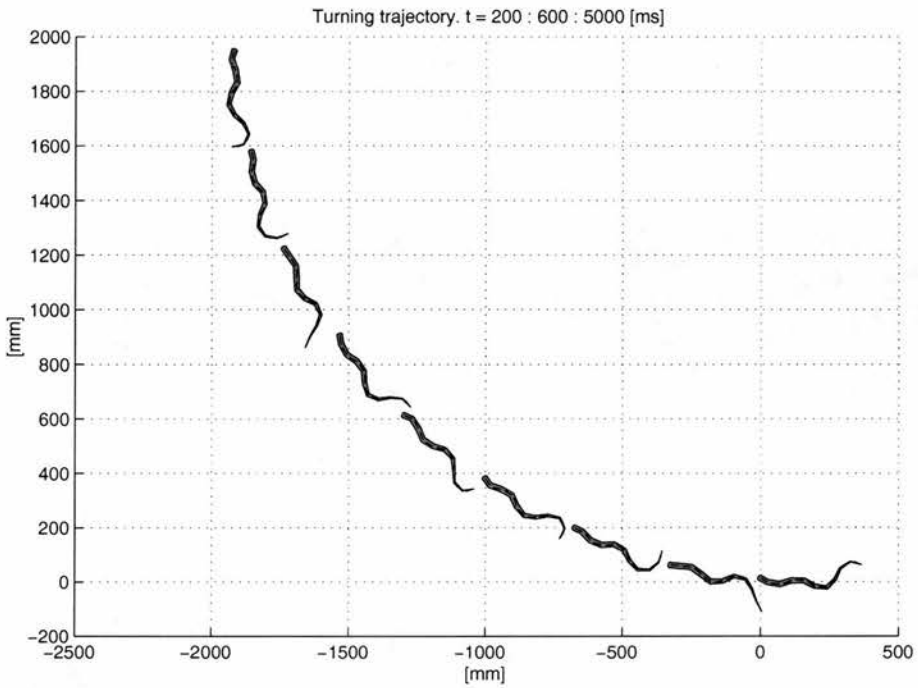


Figure 5.12: Trajectory of a turning lamprey at body scale 1.5x. The lamprey is controlled by a sinusoidal controller with amplitude = 0.925, frequency = 6 Hz and phase lag = 1.85%. Snapshots are taken from 200 ms to 5000 ms with time step of 600 ms.

speed(frequency, phase lag) surface with that obtained by Ijspeert [1998] shows that our lamprey models are behaving similarly.

- Ijspeert modified a few mechanical parameters (such as masses, coefficients of drag forces and muscle coefficients) to correct the mistakes in Ekeberg's 1993 article (the masses were too small by a factor of π and the drag coefficients were too small by a factor of 10). I use Ijspeert's model for comparison purposes. It is interesting to note that both of our implementations produce maximum speed at phase lag 2% rather than 1% as published in biological papers. It may be that when biologists talk about good swimming performance at phase lag of 1%, they do not mean the maximum swimming speed (personal communication with Hallam).

- The sinusoidal signals at each segment are generated independently by the sinu-

soidal controller. There are no intersegmental connections. The pulses of each segment have exactly the same amplitude and frequency. The only thing that is segment dependent is the onset of pulse trains. Thus, the neural waves at each segment are uniform with each other. In the case of the swimming CPG, the neural wave at each segment is affected by its neighbors due to the intersegmental couplings. Furthermore, the couplings between a specific segment and its neighbors are different depending on the location of the segments. Hence, the characteristics of the neural wave may be slightly different. As a result, even though the neural parameters are matched, there may be slight differences in performance.

- When the phase lag is large, a short mechanical wavelength results. This means that more body bending is required. Since the body has only nine joints, the model lamprey is unable to propagate a very short wavelength. As a result, the bending cannot be smooth and some of the results may be affected.

Chapter 6

Evolution of efficient swimming controllers

According to Sir James Lighthill, swimming speed and efficiency are the two qualities that fish must maintain in order to survive [Lighthill, 1970a]. If the swimming efficiency is low, the fish can quickly use up energy derived from food before they can find their next meal. The ability to maintain high swimming efficiency is especially important for lampreys because they do not eat during the long journey up-river from the sea to the breeding grounds [Williams, 1986]. Blake [1993] suggested that efficiency is a good criteria to use when comparing the swimming performance of different fish. Its increase with speed is important to the evolutionary ecology of fish. According to Webb and Kostecki [1984], selection should favor animals that can maximize rates of performance at the energy bottlenecks, as they are important for survival and reproduction. Efficiency is also important from a robotics point of view. An inefficient robotic lamprey can use up its battery power and sink in the ocean easily.

Although there are a few works on the swimming efficiency of fish [Lighthill, 1970a,b, Beamish, 1974, Webb, 1977, Webb and Kostecki, 1984, Williams, 1986, Blake, 1993, Sfakiotakis et al., 1999], the ones related to the lamprey are [Beamish, 1974, Williams, 1986]. Using efficiency as the ratio between forward swimming speed and mechanical wave speed, Williams [1986] found that the real lamprey has a high

efficiency of 0.8. So far, the details on how the neurons in the swimming CPG are coupled to produce efficient swimming is not fully understood yet.

Given its value to both the real and the artificial lamprey, in this dissertation, I use GAs to evolve efficient swimming controllers for the model lamprey originally proposed by Ekeberg [1993], with parameters corrected by Ijspeert [1998] (See Chapter 2). Two sets of experiments are conducted based on a similar fitness measure that takes efficiency into account. The first experiment involves evolving controllers with a large maximum efficiency while the second experiment involves evolving controllers with a large minimum efficiency. The reason for conducting two sets of experiments is to investigate which approach produces better results (i.e. more controllers swimming at higher efficiency). The experimental results are encouraging. Most of the evolved controllers are able to swim faster and with a higher efficiency than the corresponding prototypes (refer to Subsection 6.1.2 for an explanation of prototypes). Their speed vs. efficiency curves show that they cannot only achieve a wide range of speeds but also swim at a fairly constant efficiency (at least for speeds over 0.3 m/s). Furthermore, when using the same discrepancy function defined in Or et al. [2002] as a measurement of robustness in speed against variations in body scales, some of the controllers are more robust (have a lower discrepancy value) than the evolved controllers described in that paper.

Most importantly, the best evolved controller has achieved an efficiency of about 0.8. Thus, through the use of GA, I have found intersegmental couplings which allow the model lamprey to swim at about the same efficiency as the real one. This result, one of the key results of this thesis, could provide inspiration to biologists to gain a better understanding of the intersegmental couplings of the real lamprey.

6.1 Methods commonly used in Experiments 1 and 2

6.1.1 Efficiency calculation

There are four different definitions of swimming efficiencies [Blake, 1993, Sfakiotakis et al., 1999]. The *Froude (or propeller) efficiency* (η_p) is defined as:

$$\eta_p = \frac{T \cdot U}{P} \tag{6.1}$$

where T is the time-averaged thrust produced, U is the mean forward velocity of the fish, and P is the time-averaged power required. The measurements of these parameters are not as simple as those used in other definitions, and as a result, this definition is not commonly used. Other types of efficiency include the mechanical efficiency of the locomotor musculature (η_m), and the overall aerobic efficiency (η_a). These two types of efficiency are calculated using metabolic energy and metabolic power as parameters [Blake, 1993]. As my model does not include metabolic rate and there is no biological data for comparison, these two types of efficiency are not considered.

The one that is most commonly used is defined as the ratio of forward swimming speed (U) to backward mechanical wave speed (V) [Williams, 1986, Carling et al., 1998, Sfakiotakis et al., 1999]. Given that these parameters are easily calculated (using the fitness factors described below) and that biological data is available for comparison, I use this definition to evolve the controllers in this chapter. The efficiency and mechanical wave speed are defined as follows:

$$e = \frac{U}{V} \tag{6.2}$$

$$V = \frac{\lambda}{T} \tag{6.3}$$

where λ and T are the mechanical wavelength [m] and the mechanical period [s] respectively.

To obtain the swimming efficiency, the following routines are used:

- *CalEfficiency*: Since both the neural and mechanical simulations are not stable at the beginning of the simulations, the lamprey can end up swimming straight at any angle. The first step is to rotate the original body so that the fish is swimming parallel to the x-axis. The amount of rotation is computed in routine *FindSwimAngle* and the actual rotation is performed under *RotateBody*. Following the transformation, the mechanical periods of links 2 to 5 are found using the *FindMecPeriod* routine. If the periods of three or more of the mechanical links are defined, the mechanical wavelength can be calculated using the *FindMecWavelength* routine. Otherwise, it is set to zero. To calculate the mechanical wave speed, Equation 6.3 is used. Due to strange head and tail movements caused by the reduction in neural connections at the ends of the swimming CPG, the mechanical period of $link_2$, T_2, is used.

  Given that efficiency is defined within [0,1], and to avoid the possibility of division by zero, the values of parameters T_2, λ, U and V are checked. If any one of these variables is found to be less than or equal to zero, efficiency is set to zero. Otherwise, the efficiency can be calculated using Equation 6.2.

- *FindSwimAngle*: To obtain the swim angle (heading direction), the head positions at the time instances (t_1 and t_2) at which the same lamprey body are "in-phase" are used.[1] t_1 and t_2 correspond to the first and last pulse onsets found within the last second of the neural simulation.

- *RotateBody*: Given the state of each body link within the entire six seconds of the mechanical simulation, rotate the body in the last second so that the lamprey is swimming parallel to the x-axis. The transformed coordinates for the body links are stored in matrix TransformedBodyState. The amount of rotation is obtained from *FindSwimAngle*. The reason for rotating the body is that if the lamprey were swimming vertically across the screen, the mechanical wave crests travelling along the model lamprey body would not be detected correctly in *FindMecPeriod*.

[1] For definition of "in-phase", refer to Section 3.5.

- *FindMecPeriod*: Given the transformed body link coordinates during the last second of the mechanical simulation, scan within t_1 and t_2 for $crest_1$ and $crest_2$. $crest_1$ is the first instance at which a specific link oscillates up and down along the x-axis (nods up and down) while $crest_2$ is the second instance at which such behavior occurs. If the link under test nods less than twice during the $t_2 - t_1$ time frame, the mechanical period for this particular link is set to zero.

- *FindMecWavelength*: Given the states of the transformed lamprey body links as well as $crest_1$ and $crest_2$ of $link_2$, the mechanical wavelength can be calculated using the following method (adapted from Videler [1993]):

  Use *FindMecPeriod* to obtain the mechanical period for links 2 to 5. In other words, identify the instances at which the mechanical wave crest appears at these body links. A constraint is set in such a way that the $crest_1$ of each link appears after that of the previous link. This corresponds to the assumption that a mechanical wave is travelling from the head to the tail. If less than three mechanical periods are found, the mechanical wavelength and efficiency are both set to zero. Otherwise, use the largest possible number of these (time, link_no) data points (3 or 4 of them) to do the interpolation (more on this follows). The reason for allowing such flexibility is that from experience, the mechanical period for $link_5$ may not be found in some cases. As a result, the $crest_1$ and $crest_2$ of $link_5$ are set to zero by default. Imposing four data points to do the interpolation will lead to an incorrect interpolated line which often results in an extremely small mechanical wavelength and hence an efficiency of greater than one.

  When the mechanical period of the body links are found, the least square method is applied to interpolate a straight line (which represents the propagation of a mechanical wave along the body) to fit the available pairs of data points ($crest_1$, link_no) of the corresponding link. Using this interpolated line, the mechanical wavelength (λ) can be computed using:

  λ = (y coordinate of intersection point - 2.0) · length of a mechanical body link.

  where the length of a mechanical body link is 30 mm. For an illustration of this

method, refer to Figure 6.1.

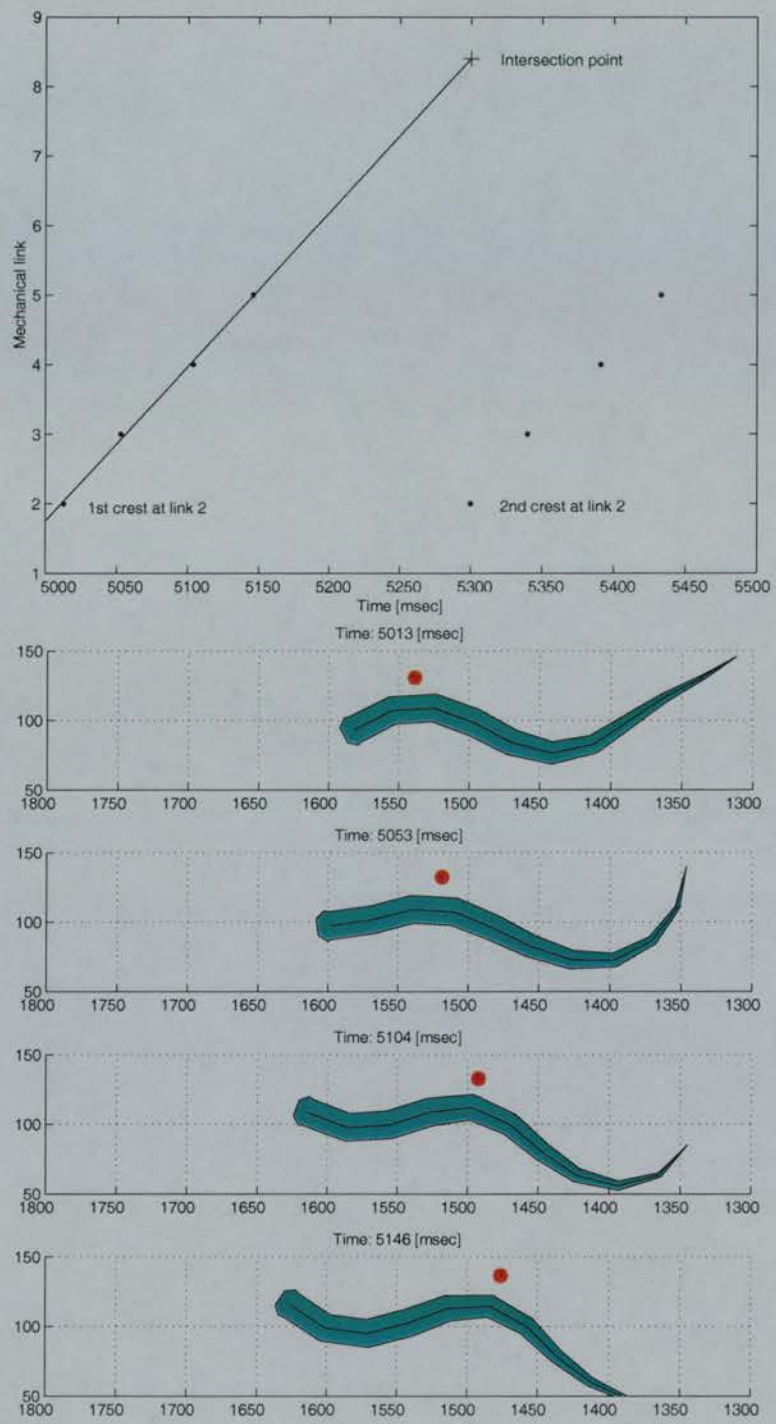


Figure 6.1: The time passage of a mechanical wave crest along the body from links 2 to 5 (*top*). The mechanical wavelength is the vertical distance from the time when the second crest appears at link 2 to the intersection across the interpolated line, multiplied by the length of a body link. The travelling of the mechanical wave crest (represented by the dot in red color) along the lamprey body from links 2 to 5 (*subsequent figures*). Data correspond to biological controller at global excitation 0.3 and extra excitation 110.0%. The amplitude, frequency and phase lag are 0.465, 3.5Hz and 1.06% respectively. The length and width of the swim-mill are in [mm].

6.1.2 Genetic Algorithms

The same real number GA described in Subsection 2.7.7 is used here. To summarize, a rank-based selection method together with two-point crossover is used. The parameters for the GA are listed in Table 6.1. Note that the number of generations is not the same across the evolutions (see Section 6.2).

Population size	40
Number of children	12
Crossover probability	0.5
Mutation probability	0.4
Mutation range	0.2

Table 6.1: GA parameters used to evolve the intersegmental couplings for the efficient swimming controllers.

Two sets of experiments are conducted. The first set is based on evolving controllers with big maximum achievable efficiency while the second set is on evolving controllers with big minimum efficiency. In each set of experiments, the following five prototype controllers are used:

1. *The Biological Controller*, hand-crafted by Ekeberg [1993] based on physiological data.

2. *Controller 2*, evolved by Ijspeert et al. [1999] using Ekeberg's segmental oscillator and intersegmental couplings evolved by GA.

3. *Controller 3*, evolved by Ijspeert et al. [1999], with both intra- and intersegmental connections evolved by GA.

4. *Hybrid Robust Controller*, hybrid segmental oscillator with the best evolved robust intersegmental couplings (both evolved by [Or et al., 2002]).

5. *Hybrid Random*, the hybrid segmental oscillator with random couplings.[2]

[2]For the reason why this controller is included, refer to Section 6.4.

For each prototype controller, six evolutions (runs) are performed. This makes a total of 30 runs for each set of experiments. Note that the initial population of each run contains a prototype controller together with 39 other randomly generated individuals.

For interested readers, their connection matrices, efficiency surfaces and the speed vs. efficiency scatter plots are included in Appendix A.

6.1.3 Fitness calculation

The fitness function used to evolve efficient swimming controllers is defined as follows. For definitions of fitness factors (*min_fit_oscil*, *fit_lagcontrol*, *fit_speed*) that were derived in Ijspeert [1998], refer to Subsection 2.7.7.

$$Fitness3 = min\_fit\_oscil \cdot fit\_lagcontrol \cdot fit\_freqcontrol \cdot fit\_speed \cdot fit\_efficiency \quad (6.4)$$

where

- $fit\_freqcontrol = \begin{cases} 0.05 + \frac{freq\_range2}{1+phase\_lag\_range2} & \text{if} < 1 \\ 1 & \text{otherwise} \end{cases}$

  Although this fitness factor looks the same as the one designed in Ijspeert [1998], I have made several changes to adapt to the evolutions of efficient controllers. Recall that neural simulations with different global excitations and extra excitation fixed at *extra0* are conducted when this fitness factor is called. Since the calculation of swimming efficiency requires both neural and mechanical simulations, *fit_speed* and *fit_efficiency* are called within this routine. In order to discourage the evolution system from producing controllers which generate irregular pulses (causing extremely low mechanical wavelength and hence efficiency greater than one), *min_fit_oscil* is called at the end of each neural simulation within this routine to check for pulse regularity. If any segment is found to have generated irregular pulses, the corresponding swimming speed and efficiency are both set to zero. Otherwise, mechanical simulation is conducted for speed and

efficiency calculations. The results are recorded in a table along with the corresponding neural characteristics. This procedure allows the *fit_efficiency* fitness factor to search for either the maximum or minimum efficiency (depending on the experiment) without repeating the neural simulations.

- $fit\_efficiency = \begin{cases} 0.95 \cdot \frac{max\_efficiency - G}{G - B} + 1.0 & \text{if} < 1 \\ 1 & \text{otherwise} \end{cases}$

max_efficiency is the maximum efficiency. The bad boundary B is 0 and the good boundary G is 1.

To find the *max_efficiency*, the routine searches the table created in *fit_freqcontrol* for the maximum efficiency. If there is no valid entry (the range of oscillation frequency is not defined), the maximum efficiency is set to zero. Otherwise, the value is linear transformed to a value between 0.05 and 1. Note that the difference between experiments one and two is that in the latter case, the variable *max_efficiency* is replace by *min_efficiency*. The routine searches for the minimum non-zero efficiency value instead of the maximum efficiency.

6.2 Results

The fitness factors were tested and tuned up during the evolutions. As a result, I cannot compare the evolutions in terms of the number of generations required to achieve similar fitness. However, the efficiency of the fittest individuals across different evolutions can still be compared.

I monitored the progress of the evolution weekly. Some of the most fit individuals from these evolutions were tested. When the fitness of these individuals stopped increasing significantly, I stopped the evolutions (after 2 months). Using the procedure described in Section 3.6, the best individual of each run is tested over a range of global and extra excitations to determine the ranges of amplitude, frequency, phase lag, speed and efficiency which it can achieve. The corresponding surfaces are plotted for com-

parison. Note that efficiency values obtained under excitation combinations that lead to irregular neural waves (i.e. $min\_fit\_oscil < 0.45$) are filtered out before the efficiency surfaces are plotted. From the empty regions found in the efficiency surface, one can relate the abnormal regions (such as holes and rough regions) found in other surfaces.

Due to space limitations, the results for only 20 of the 60 evolved controllers are presented here. The criteria for choosing these controllers is a balance between high swimming efficiency and high fitness. Based on these criteria, two controllers from each prototype group are chosen for comparison. Since there are five prototypes and two sets of experiments, this makes a total of 20 controllers.

The results for the two set of experiments are presented in the following subsections. For details of the neural configuration and performance surfaces of each controller, refer to Appendix B to E.

6.2.1 Results of Experiment 1: On evolving controllers with big maximum efficiency

The results for the 10 selected controllers based on the bigmax approach are summarized in Table 6.2.

The table indicates that except for the run6 controller and those evolved with the hybrid segmental oscillator as the prototype (the bottom four controllers), the rest of the evolved controllers can achieve a higher maximum efficiency than the corresponding prototypes. Among the 10 evolved controllers, the run12 controller has the highest efficiency value of 0.86.

6.2.2 Results of Experiment 2: On evolving controllers with big minimum efficiency

The results for the 10 selected controllers based on the bigmin approach are summarized in Table 6.3. The table indicates that except for run20, run21 and run30 controllers (again all evolved with the hybrid segmental oscillator as the prototype), the rest of the evolved controllers are more efficient than their corresponding prototypes. Among the 10 evolved controllers, the run40 controller has the highest efficiency value

	Fitness	Amplitude range	Frequency range in [Hz]	Phase lag range in [%]	Speed range in [m/s]	Efficiency range
biological	0.00	[0.0, 0.8]	[1.6, 5.5]	[-0.1, 1.7]	[-0.09, 0.45]	[0.05, 0.58]
run5	0.11	[0.2, 0.8]	[1.6, 7.2]	[-2.9, 2.6]	[-0.09, 0.51]	[0.05, 0.61]
run32	0.10	[0.0, 0.8]	[1.3, 5.7]	[-1.4, 3.2]	[-0.03, 0.53]	[0.05, 0.64]
controller 2	0.00	[0.0, 0.8]	[1.7, 6.0]	[-3.1, 3.2]	[-0.09, 0.49]	[0.02, 0.60]
run6	0.13	[0.0, 0.8]	[1.4, 7.5]	[-2.6, 2.8]	[-0.09, 0.52]	[0.05, 0.59]
run10	0.11	[0.0, 0.8]	[1.6, 5.7]	[-2.2, 3.4]	[-0.03, 0.50]	[0.01, 0.63]
controller 3	0.00	[0.0, 0.6]	[1.3, 5.5]	[-0.2, 1.9]	[-0.08, 0.43]	[0.06, 0.58]
run12	0.10	[0.0, 0.6]	[1.4, 6.4]	[-2.3, 8.6]	[-0.07, 0.49]	[0.02, 0.86]
run15	0.06	[0.0, 0.6]	[1.5, 5.9]	[-0.0, 1.8]	[-0.08, 0.44]	[0.03, 0.64]
hybrid robust	0.11	[0.0, 0.8]	[1.8, 7.1]	[0.0, 3.1]	[-0.02, 0.49]	[0.08, 0.69]
run21	0.15	[0.0, 0.8]	[1.4, 7.1]	[-0.0, 2.8]	[-0.03, 0.48]	[0.18, 0.68]
run23	0.11	[0.0, 0.8]	[1.2, 7.1]	[0.0, 3.3]	[-0.02, 0.48]	[0.30, 0.61]
hybrid random						
run20	0.05	[0.0, 0.8]	[1.5, 7.6]	[-1.3, 2.9]	[-0.05, 0.48]	[0.07, 0.65]
run27	0.09	[0.0, 0.7]	[1.5, 7.0]	[-0.4, 2.2]	[-0.02, 0.38]	[0.07, 0.62]

Table 6.2: Summary of results for the evolved efficient controllers in Experiment 1. The table lists the performance of the best individual from each evolution. The evolution is based on the bigmax approach. Note that the hybrid random prototype generates irregular neural waves due to random couplings. As a result, all the parameters are undefined.

of 1.03.[3]

6.3 Inherited properties: Robustness of the evolved controllers

This section addresses two types of robustness inherited by evolving efficient controllers: 1) robustness in swimming efficiency against variations in speed, and 2) robustness in swimming speed against variations in body scales (addressed in [Or et al.,

[3]Efficiency greater than one is impossible. This value is caused by the breakdown of the wavelength calculation algorithm. Thus, this value is later considered to be invalid (refer to Section 6.4 for details). Note that after this value is eliminated, the highest efficiency the run40 controller can achieve is 0.67.

	Fitness	Amplitude range	Frequency range in [Hz]	Phase lag range in [%]	Speed range in [m/s]	Efficiency range
biological	0.00	[0.0, 0.8]	[1.6, 5.5]	[-0.1, 1.7]	[-0.09, 0.45]	[0.05, 0.58]
run3	0.11	[0.0, 0.8]	[1.6, 5.6]	[0.0, 6.4]	[-0.05, 0.48]	[0.05, 0.68]
run29	0.06	[0.2, 0.8]	[1.6, 5.5]	[-0.7, 1.8]	[-0.16, 0.47]	[0.05, 0.68]
controller 2	0.00	[0.0, 0.8]	[1.7, 6.0]	[-3.1, 3.2]	[-0.09, 0.49]	[0.02, 0.60]
run9	0.09	[0.0, 0.8]	[1.6, 5.5]	[-0.3, 2.1]	[-0.15, 0.51]	[0.06, 0.76]
run10	0.08	[0.0, 0.8]	[1.6, 5.5]	[-1.7, 3.1]	[-0.03, 0.48]	[0.03, 0.70]
controller 3	0.00	[0.0, 0.6]	[1.3, 5.5]	[-0.2, 1.9]	[-0.08, 0.43]	[0.06, 0.58]
run11	0.06	[0.1, 0.6]	[1.3, 7.9]	[-3.1, 3.3]	[-0.05, 0.48]	[0.03, 0.59]
run13	0.06	[0.0, 0.6]	[1.3, 6.9]	[0.0, 2.4]	[-0.08, 0.48]	[0.03, 0.58]
hybrid robust	0.06	[0.0, 0.8]	[1.8, 7.1]	[0.0, 3.1]	[-0.02, 0.49]	[0.08, 0.69]
run21	0.09	[0.0, 0.8]	[1.2, 7.0]	[-0.1, 3.5]	[-0.02, 0.49]	[0.02, 0.61]
run40	0.09	[0.0, 0.8]	[1.5, 7.1]	[0.0, 3.6]	[-0.05, 0.46]	[0.12, 1.03]
hybrid random						
run20	0.10	[0.0, 0.8]	[1.3, 7.0]	[-1.1, 5.3]	[-0.07, 0.46]	[0.23, 0.68]
run30	0.20	[0.0, 0.7]	[2.0, 7.1]	[0.0, 3.1]	[-0.02, 0.48]	[0.05, 0.63]

Table 6.3: Summary of results for the evolved efficient controllers in Experiment 2. The table lists the performance of the best individual from each evolution. The evolution is based on the bigmin approach. Note that the hybrid random prototype generates irregular neural waves due to random couplings. As a result, all the parameters are undefined.

2002]). Other types of robustness (such as swimming speed and efficiency as well as neural characteristics against random variation in neural connections) will be addressed in Chapter 7.

Robustness in efficiency is important from a robotics point of view. We want a robotic lamprey which can both achieve a wide range of swimming speed and also be able to maintain high efficiency. It is unavoidable that the robot changes its swimming speed during the course of its mission (such as when it changes its swim direction). If it were unable to maintain efficiency, its internal power could run out before it reaches its destination.

Robustness in swimming speed is also important to both real and model lampreys. As mentioned in Chapter 4, it is critical for the robotic lamprey to maintain swimming

speed even when the body is subjected to physical variations due to craftsmanship or design. Robustness against variations in body parameters is also important from an adaptive behavior viewpoint because as the fish grows, its body characteristics change. A locomotion controller which can tolerate such variations is essential to the survival of the being.

6.3.1 Robustness in swimming efficiency against variations in speed

In order to determine the relationship between the swimming speed and efficiency of the evolved controllers, I have plotted the efficiency vs. speed curves of these controllers for speed ranges from 0.05 to 0.6 m/s in steps of 0.05. As the same speed can correspond to more than one efficiency value, I have chosen the points which have relatively high efficiency values, yet which are able to fit with the neighboring data points. The reason for choosing high efficiency values which fit the neighboring data points is that a controller which can achieve high efficiency but only at an extremely small range of speeds is not very useful. The curves are shown together with the corresponding performance surfaces in Appendix B and C. As each curve only represents a possible trend, the scatter plots of normal body scale are included as a reference. In order to compare how this relationship varies with body scales, scatter plots for lamprey with body scales at 0.8x and 1.2x are also provided (see Appendix D and E).

The figures show that the relation between speed and efficiency can be classified into two types, namely:

- efficiency increases with speed
- efficiency increases with speed initially and then stays fairly constant (for speed over 0.3 m/s)

6.3.2 Robustness in swimming speed against variations in body scales

In order to determine whether evolving swimming controllers based on efficiency can indirectly bring in robustness in swimming speed against variations in body scales, I

used the discrepancy equation (refer to the end of Subsection 4.5.5 for definition) to calculate their discrepancy. The results for the selected controllers evolved under the bigmax and bigmin approaches are summarized in Table 6.4 and Table 6.5 respectively. For comparison purposes, the discrepancy of the best five evolved robust controllers obtained in Or et al. [2002] are shown in Table 6.6.

Controller	Discrepancy
biological	0.318
run5	0.240
run32	0.205
controller 2	0.223
run6	0.238
run10	0.243
controller 3	0.330
run12	0.253
run15	0.322
hybrid robust	0.258
run21	0.218
run23	0.282
hybrid random	undefined
run20	0.108
run27	0.260

Table 6.4: Comparison of the prototype controllers with the corresponding evolved efficient controllers in terms of discrepancy (robustness against variation in body scales). The evolved controllers are from Experiment 1 based on the bigmax approach.

Table 6.4 indicates that under the bigmax approach, five of the evolved controllers (run5, run32, run12, run15 and run21) have less discrepancy than their corresponding prototype. The other three controllers (run6, run10 and run23) have discrepancy just slightly over that of their prototypes. As for the remaining two controllers (run20 and run27), there is no basis of comparison as the prototype controller (hybrid segmental oscillator with random couplings) cannot swim at any scale.

Compared with the evolved controllers obtained in Or et al. [2002], the run20 controller has the least discrepancy and therefore is more robust than those evolved with a

Controller	Discrepancy
biological	0.318
run3	0.289
run29	0.195
controller 2	0.223
run9	0.257
run10	0.164
controller 3	0.330
run11	0.204
run13	0.214
hybrid robust	0.258
run21	0.329
run40	0.345
hybrid random	undefined
run20	1.367
run30	0.385

Table 6.5: Comparison of the prototype controllers with the corresponding evolved efficient controllers in terms of discrepancy (robustness against variation in body scales). The evolved controllers are from Experiment 2 based on the bigmin approach.

fitness measure that takes robustness into account.

Table 6.5 indicates that five of the evolved controllers (run3, run29, run10, run11 and run13) have lower discrepancy values than their corresponding prototypes. The run9 controller has discrepancy slightly higher than its prototype (controller 2). The two controllers (run21 and run40) evolved with the hybrid robust controller as the prototype have higher discrepancy than their prototype. Again, there is no basis of comparison for the run20 and run30 controllers as their prototype (the hybrid random controller) cannot swim at any scale. Compared with the evolved controllers obtained in Or et al. [2002], the run10 controller evolved in this chapter has the lowest discrepancy value.

These observations indicate that evolving controllers based on efficiency can indirectly lead to robustness in swimming speed against variations in body scales. It is interesting to note that the bigmax run20 controller has the lowest discrepancy value of

Controller	Discrepancy
biological	0.318
run30	0.198
hybrid robust	0.258
run5	0.211
run9	0.330
run10	0.357

Table 6.6: Summary of discrepancy values for the best evolved robust controllers obtained in [Or et al., 2002]. Note that the runs presented in this table have no relation to those mentioned in this chapter. In addition, all the discrepancy values listed in the table are computed using speeds obtained through the new speed calculation algorithm.

0.108. Compared with all the hybrid segmental based controllers, this value is at least a half of those evolved in this chapter and about a third of some of the robust controllers evolved in [Or et al., 2002]. In terms of brainstem modulation of the neural waves and swimming speed, the efficient controllers are better (have smoother surfaces) than those corresponding to robust controllers.

Hence, the fitness function proposed in this chapter is also a better choice for evolving controllers which are robust against changes in body scales.

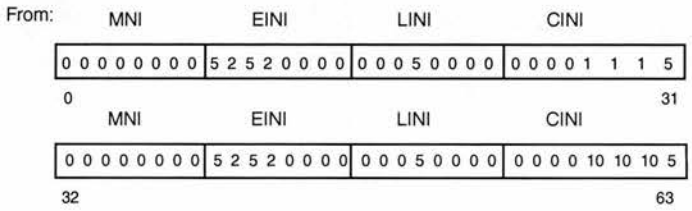
6.4 Discussion of the methods

In this section, I provide empirical justification for using the GA to evolve efficient swimming controllers. This is followed by a discussion on implementation issues.

6.4.1 Empirical justification for using the GA to evolve efficient swimming controllers

In order to investigate whether the evolved efficient controller (run9 under the bigmin approach) is near a local optimum, I applied the classical optimization technique named "hill climbing" to the chromosome which encodes the intersegmental couplings of the swimming neural networks. Given that there are 64 genes in total, it would be

possible to mutate them individually and observe how fitness changes. However, after careful observation, I found that mutating genes at positions 0 to 7 and 32 to 39 had no effect on the behavior of the controller, and hence the fitness. The reason for this is that genes at these positions correspond to neural connections from the motoneurons to the interneurons. These connections are not considered in the program as feedback is not implemented. (Refer to Figure 6.2 for the encoding of intersegmental couplings.)



From / To	MNI	EINI	LINI	CINI	CINr	LINr	EINr	MNr
MNI	0	5	0	0	5	0	0	0
EINI	0	2	0	0	1	0	0	0
LINI	0	5	0	0	1	0	0	0
CINI	0	2	5	0	1	0	0	0
CINr	0	0	0	1	0	5	2	0
LINr	0	0	0	1	0	0	5	0
EINr	0	0	0	1	0	0	2	0
MNr	0	0	0	5	0	0	5	0

Rostral

From / To	MNI	EINI	LINI	CINI	CINr	LINr	EINr	MNr
MNI	0	5	0	0	5	0	0	0
EINI	0	2	0	0	10	0	0	0
LINI	0	5	0	0	10	0	0	0
CINI	0	2	5	0	10	0	0	0
CINr	0	0	0	10	0	5	2	0
LINr	0	0	0	10	0	0	5	0
EINr	0	0	0	10	0	0	2	0
MNr	0	0	0	5	0	0	5	0

Caudal

Figure 6.2: Encoding of intersegmental couplings. The couplings of each neuron to its neighbor in both the rostral and caudal directions are stored in two separate matrices. A chromosome of length 64 is used to encode the information. Elements from the first 32 genes correspond to connections stored in the matrix "Rostral". The last 32 genes correspond to connections stored in the matrix "Caudal". Note that for illustration purposes, the number of extensions represented by each gene has not been transformed to a real number between 0 and 1 in the figure.

To mutate a gene, one varies its value in terms of either a percentage or an absolute value. (Recall that each gene is a real number between 0 and 1.) The mutated gene value is then scaled by a factor of 12 (the maximum length of couplings in the original transformation) and then rounded to the closest integer to represent the length of

intersegmental couplings among the source and target neurons. In order to decide on the method for mutating the genes, I compared the effect of mutation under these two methods, as shown in Tables 6.7 and 6.8.

Gene value	0 [%]	10 [%]	20 [%]	30 [%]	40 [%]	50 [%]	60 [%]	70 [%]	80 [%]	90 [%]	100 [%]
0.013	0	0	0	0	0	0	0	0	0	0	0
0.5	6	7	7	8	8	9	10	10	11	11	12
1.0	12	13	14	16	17	18	19	20	22	23	24

Table 6.7: The effect of mutating gene value by percentage. Note that each table element represents the length of the intersegmental connection after mutation and scaling.

Gene value	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0.013	0	1	3	4	5	6	7	9	10	11	12
0.5	6	7	8	10	11	12	13	14	16	17	18
1.0	12	13	14	16	17	18	19	20	22	23	24

Table 6.8: The effect of mutating gene value by an absolute value. Note that each table element represents the length of the intersegmental connection after mutation and scaling.

Table 6.7 shows that mutating a value close to zero in terms of percentage has no effect on the number of intersegmental couplings. Hence, the behavior of the controller and its fitness are not affected. Since a gene value near zero is possible (0.013 is chosen from the chromosome for the evolved efficient controller as a realistic example), I used the second method. I varied each gene individually from -1.0 to 1.0 in steps of 0.1. The effect of each mutation on the total fitness is recorded. Note that as the gene is bounded by 0 and 1, any mutated value outside this range is set to the closest end.

Since it is impossible to visualize a multi-dimensional fitness landscape, I plotted the fitness value at different amounts of mutation for each gene position under investigation (see Figures 6.3 to 6.8). The figures indicate that there are several types of one-dimensional landscape. They can be classified as *flat, smooth plateau, steep mountain edge, roughed surfaces and spikes (multi-peaks)*. An example of these can be found in curves corresponding to mutations at gene positions 12, 8, 10, 11 and 19 respectively.

Figure 6.3: The effect of mutating genes 8 to 12 on the total fitness. Counter-clockwise from the top-left: mutating the neural connection (in the rostral direction) from the EINI neurons to the MNI, EINI, LINI, CINI and CINr neurons.

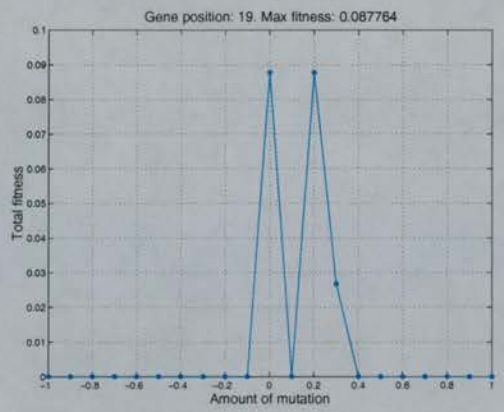


Figure 6.4: The effect of mutating gene 19 on the total fitness. Mutating the neural connection (in the rostral direction) from the LINl neurons to the CINl neurons.

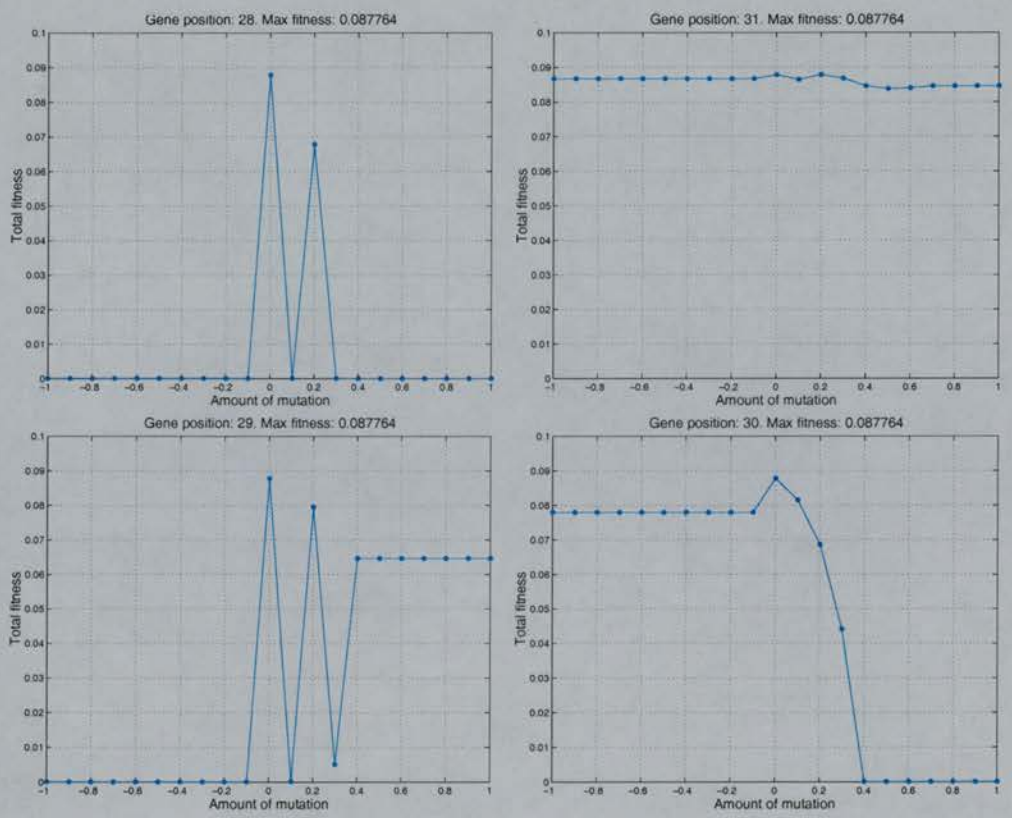


Figure 6.5: The effect of mutating genes 28 to 31 on the total fitness. Counter-clockwise from the top-left: mutating the neural connection (in the rostral direction) from the CINl neurons to the CINr, LINr, EINr and MNr neurons.

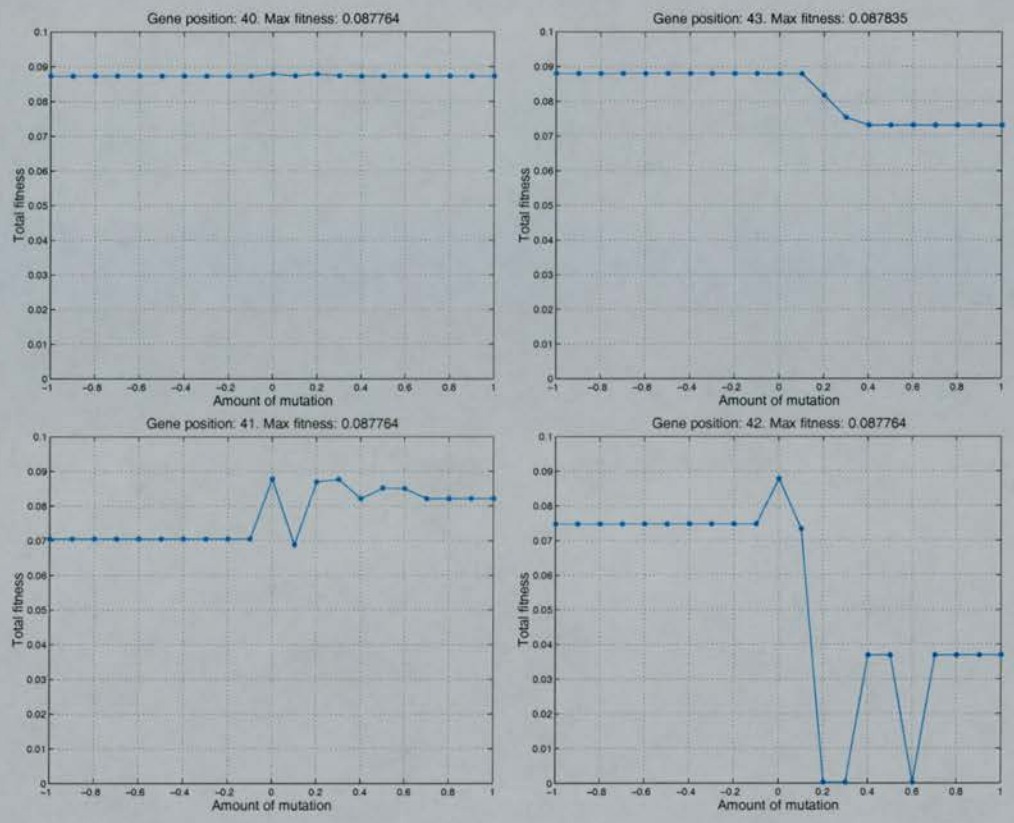


Figure 6.6: The effect of mutating genes 40 to 43 on the total fitness. Counter-clockwise from the top-left: mutating the neural connection (in the caudal direction) from the EINI neurons to the MNI, EINI, LINI and CINI neurons.

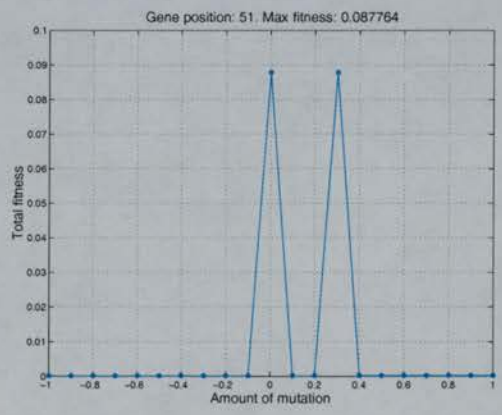


Figure 6.7: The effect of mutating gene 51 on the total fitness. Mutating the neural connection (in the caudal direction) from the LINI neurons to the CINI neurons.

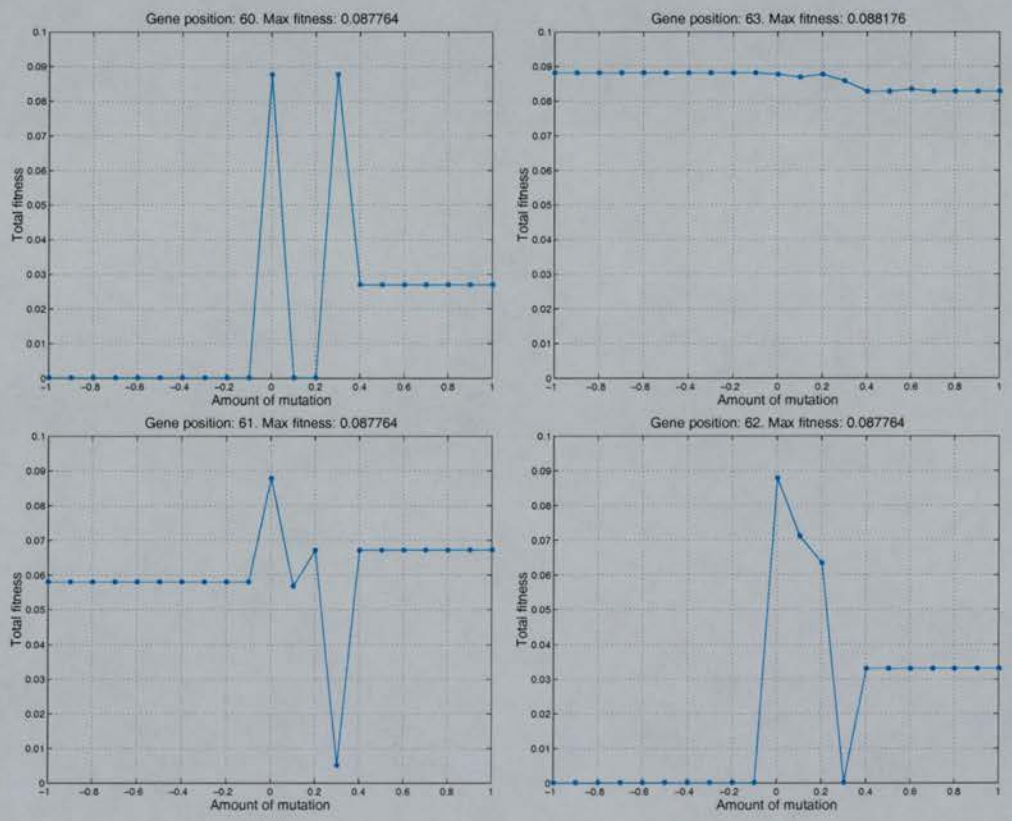


Figure 6.8: The effect of mutating genes 60 to 63 on the total fitness. Counter-clockwise from the top-left: mutating the neural connection (in the caudal direction) from the CINl neurons to the CINr, LINr, EINr and MNr neurons.

A comparison of these curves shows some interesting features of the landscape. For instance, the total fitness remains fairly constant for connections to the motoneurons. For connections from the EINl neurons to the LINl neurons, mutating the gene by a small amount causes the fitness to drop to zero rapidly. For connections from the EINl to EINl neurons, the fitness varies by less than 0.002. Similar observations can be made for neural connections from the EINl to CINl neurons. As for the burst terminating LINl neurons, their connections to the CINl neurons have a large effect on fitness (oscillating rapidly between 0 and the maximum fitness). This can also be found for the connections from the CINl neurons to the contralateral CINr, LINr and EINr neurons. It is interesting that all of the above relationships hold for couplings in both rostral and caudal directions. Recall that (based on both real biological and modelling experiments) the connections from the CIN neurons have an effect on the phase lag. Given that the swimming efficiency is calculated using the mechanical wavelength (which is dependent on the phase lag), mutating genes corresponding to connections from the CIN neurons should have a huge impact on fitness. The figures show that this is indeed the case.

Given that the GA is exploring different regions of the fitness landscape in parallel, it is less likely to be trapped in local optima in multi-peak regions (refer to the curve with mutations at gene position 28 for an example). Note that except for curves corresponding to mutation at gene positions 8, 43 and 63, the maximum fitness of the rest is 0.087764. This is the same as that of the evolved efficient controller. Among the three curves, the fittest is 0.088176 (achieved through mutation at gene 63) which is just slightly higher than the rest. Thus, the evolved efficient controller is near a local optimum and so better evolved controllers could still be found. Note that there is no way of telling whether this local optimum is also a global optimum as it is possible that higher fitness could be achieved through mutations at several gene positions simultaneously.

The behavior of the fitness function as a function of generation for the evolution of the efficient controller is shown in Figure 6.9. The figure indicates that at generation 60 (where I stopped the GA to evaluate the evolved efficient controller), there are upward trends for both the minimum and average fitness. The maximum fitness only recently had a substantial increase. It is thus still possible that better controllers could

be found. However, a later experiment shows that even when I increased the number of generations to 150, the maximum fitness improves only by about 8% (from 0.0878 to 0.0948). From the figure, it seems that the three fitness curves start to converge at generation 150. However, there is not enough evidence to say whether this is in fact the case. Given that the evolved efficient controller is able to produce a higher speed and efficiency than any other controller, and that at the time of the original experiments, there was no way of telling when or whether the maximum fitness will ever increase significantly, I stopped the GA at generation 60.

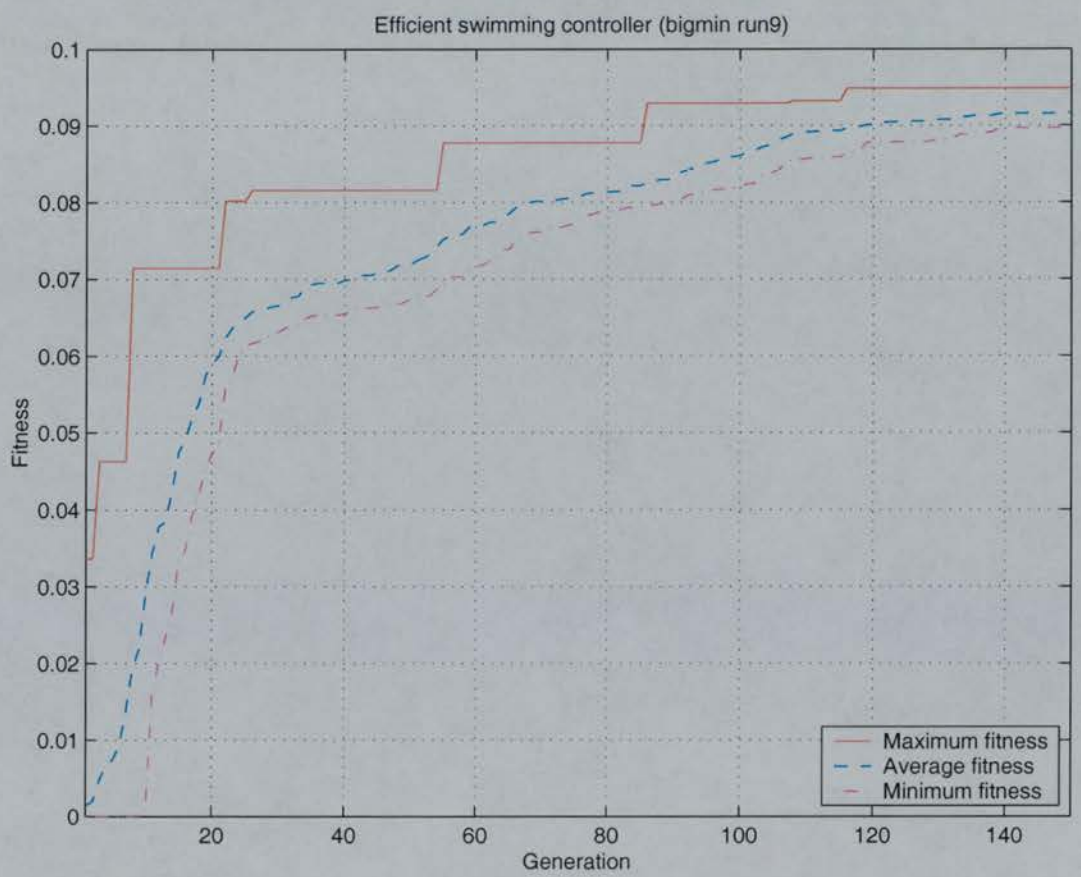


Figure 6.9: Fitness as a function of generation for the evolution on efficient swimming controller.

In this chapter, my aim is to optimize a controller which can produce high frequency of oscillation, large phase lag range, as well as high swimming speed and efficiency. Given that it takes approximately two months to evolve the efficient swim-

ming controller and that the evolution had still not converged when I terminated it, one might suspect whether a classical optimization technique [4] might work better than the GA. According to [Adby and Dempster, 1974, Goldberg, 1989, Deb, 2002], classical optimization techniques can be divided into two groups, namely the direct search method and the gradient-based method. The former approach considers only an objective function to direct the search while the latter requires the derivatives of the objective function (which may not even exist) as additional information. In the discussion which follows, I compare these optimization techniques and show that the GA, although it might not be universally the most efficient method, is still a more appropriate tool for finding an efficient swimming controller in this case.

In a simple search method, one adjusts each parameter in turn for a higher objective function value until no further improvement can be made. If the function is affected strongly by the interactions of several parameters, as in the case of the efficient controller,[5] the results of such adjustments can be complicated. This kind of one-at-a-time-search is very inefficient for complicated functions because the direction of movement often does not correspond to the direction of the optimum. Given this limitation, the simplex algorithm was developed to define the direction of movement based on a local search. According to Adby and Dempster [1974], the simplex search algorithm is carried out by selection of $n+1$ mutually equidistant search points which define a simplex. With respect to the problem of searching for an efficient swimming controller, it will be a $n = 64$ dimensional version of the tetrahedron in space. At each step, a new simplex is formed by reflecting the vertex that gives the maximum value of the objective function through the remaining n points. This approach suffers from two problems. First, the locations of the initial vertices used to form the first simplex have an effect on the quality of the solution. (Consider two hill climbing paths separated by a deep and wide valley; the path up on hill leads to global optimum while the other traps the climber at the top of a local peak.) Second, the size of the simplex can affect both the result and the speed of finding a local optimum. For instance, suppose that a

[4] Deb [2002] defines a classical optimization technique as any search and optimization algorithm which starts from a random initial guess solution and uses a deterministic transition rule to update a single solution at each iteration.

[5] Specific combinations of intersegmental couplings are required to coordinate the lamprey movement to achieve high swimming efficiency.

high peak is located near the initial vertices. If the distance between each point is too great, this peak could be missed. On the other hand, if we start the search with points very close to each other, it can take longer for the search to converge to an optimum solution.

Nelder and Mead [1965] proposed a variant of the simplex method. Their improved algorithm allows adaptive change in the distance between the points which form the simplex. According to Adby and Dempster [1974], this method is very effective in finding the optimum and is "roughly comparable in efficiency to more advanced methods for problems in many dimensions, say $n > 100$." Hence, the Nelder and Mead approach could possibly be applied to find an efficient swimming controller in less time than GA. However, as I partially showed earlier, the fitness landscape which can not be known analytically is rugged. Due to the tendency of the simple search method to get stuck at local optima, the performance of the resulting controller may not be as good as that created by the GA.[6] Other search methods include the exhaustive search and the random search. In the exhaustive search, the algorithm is like an explorer who walks through the fitness landscape in a regular way until the entire search space has been explored. Although this method can lead to the best solution (global optimum), it is not practical, especially when the search space is huge (this has been called the *curse of dimensionality* problem). In the random search, on the other hand, individual search points (controllers) are chosen randomly for fitness testing. Given that the search space is huge (the length of the chromosome is 64), it is unlikely that an optimal solution can be found within a reasonable time.

In terms of the calculus-based gradient methods, experimental results show that they can be better than the search methods [Adby and Dempster, 1974]. However, they require that the objective function be a continuous differentiable. This is often simply not the case in real-world problems. The situation worsens if local landscape is flat. In such a case, there is no local gradient to guide the search, or else it may lead the search

[6] According to Langdon and Poli [2002], the initial location of the search is very important, as the global optimum may never be reached from that position. In GAs, instead of doing the local search once in a run, a population of explorers are spreaded randomly across the landscape. Those that have reached relatively high points (fitness) are rewarded by being allowed to have children who may replace the explorers with low fitness. Thus, the GAs are less likely to be trapped in local optima and several optimal solutions can be found in a single run (generation).

along an unnecessarily long path to the local peak. A variant of the gradient approach called Generalized Reduced Gradient (GRG) assumes that the objective function is smooth and nonlinear. It is very accurate and fast (often 10 to 20 times faster than the GA) in finding the local optimal (http://www.frontsys.com/pril1.htm). Unfortunately, as I have shown earlier, the fitness landscape I am dealing with have many optima. Furthermore, the gradient (which may not even exist) cannot be found analytically. Even if this is possible, a lot of time will be required to estimate it by sampling. Therefore, the GRG and the other gradient methods are not suitable as a search method here.

When one considers using a classical optimization technique to replace the GA as a search method, one should bear in mind that the fitness function used to evolve the efficient controller is composed of several fitness factors, some of which might be in conflict with each other. For example, while the fitness factor *fit_lagcontrol* encourages solutions which can produce a large phase lag, it does not necessarily agree with *fit_speed*. For instance, a large phase lag does not necessarily allow the lamprey to swim in such a way as that the lateral forces along the body cancel out to achieve high swimming speed. Since there is no single solution which can be the optimal solution to such multiple conflicting fitness function, several trade-off optimal solutions result. The classical optimization techniques mentioned above become undesirable as they can at best find only one solution in a run (and that solution is often only a local optimum). With a population-based approach such as the GA, several optimal solutions can be found in a single run, and the GA is therefore more efficient as a search method. Of course, if the optimization problem has only a single optimum, individuals in the GA population are expected to converge to this optimum. In such case, a classical optimization technique would be more efficient than the GA because at each iteration, only one search point (rather than a set of points) is evaluated.

Finally, although it appears that the GA has taken two months to find the efficient controller and that other classical optimization techniques (such as hill-climbing and the calculus-based search) might have done the optimization in minutes or hours, one should bear in mind that it is not the GA itself that takes up the time. Rather, it is the evalutaions of the fitness function. Recall in Subsection 6.1.3 that to assign a fitness value to an individual controller, its swimming performance is tested at differ-

ent excitation combinations. The desired controller is one which has its frequency of oscillation and swimming speed increase smoothly with global excitation while independently, its phase lag has to be increased with extra excitation. The bottleneck in the evolution is the time it takes to complete one neuromechanical simulation. One may consider shortening both the neural and mechanical simulations by running the simulations with shorter simulated seconds, but then the swimming speed calculations become less accurate. In order to compute the swimming speed correctly, 8 seconds of neural simulation followed by 6 seconds of mechanical simulation is necessary. (For more details, refer to Subsection 3.6.) An alternative to save time is to test the controller at coarse excitation grids. Although this does reduce the number of neuromechanical simulations, the quality of the solutions will be decreased (i.e. the controllers are less likely to produce smooth performance surfaces across a wide range of global and extra excitations.) One encouraging point worth mentioning is that at the time of this writing, the speed of the fastest desktop PC is at least 6 times faster then the SUN workstation I used to evolve the efficient controller a year ago. In other words, it would now take about 9 days instead of 2 months to complete the same evolution!

Earlier in this subsection, I gave partial evidence that the fitness landscape for evolving efficient swimming controllers contains multiple optima. Given that the GA is searching for good solutions at different parts of the huge search space in parallel (*implicit parallelism*) to avoid getting stuck at local optima, and that it does not require the objective function to be continuous and differentiable, it is more appropriate than the classical optimization methods as a design tool for finding the efficient swimming controllers.

Note that it might be possible to create an algorithm which can find an efficient controller faster than the GA. For example, one could generate a large initial population of, say, 100 randomly generated individuals and then perform local hill-climbing for each of them. To avoid the problem of bad starting points, 10 sets of experiments can be conducted. However, this approach will not take significantly less time because a larger population size is required to search different regions of the landscape. Also, the performance of each controller at each excitation combination still has to be evaluated. Since that there are many optima in the search space, it is possible that a swimming

controller could be found. However, the best created controller may not be as good as the evolved efficient controller. Future experiments are required for future comparison.

Given the comparison above, I conclude that a population-based search method is the most appropriate one to use for solving the problem of finding an efficient swimming controller, in spite of the fact that, in general, the GA may not be the most efficient technique available.

6.4.2 Discussion of implementation

Based on my three month's experience of evolving robust swimming controllers (described in Or et al. [2002]), where all individuals were random in the initial populations, I included a prototype controller in each initial population to guide the GAs to search for regions of possible solutions (controllers that can at least swim) in the search space. Although this approach can reduce the amount of time needed to evolve efficient controllers, there is the possibility that all the evolved controllers (under the same prototype) end up similar to each other. Fortunately, this did not pose a serious problem in my case. Most of the evolved controllers (even evolved with the same prototype) have different neural configurations and performance surfaces. Thus, I obtained good solutions (controllers which are able to make the model lamprey swim like the real one, with oscillation frequency increasing with global excitation and phase lag increasing with extra excitation) in a shorter time by luck. The biological controller, controller 2 and controller 3 prototypes all have fitness very close zero under the new fitness function. As a result, they could not dominate the entire population. As for the hybrid robust prototype controller, it has a fitness of 0.11 and 0.06 in experiments one and two respectively, which is relatively low. The mutation and crossover operators of GAs could easily move the search to neighbouring regions. Evolutions based on the hybrid random prototype were included just in case my approach failed. In general, if there were plenty of time and computing resources, it might be better to have an initial population with all randomly made individuals. This allows more different types of controllers to be evolved.

In Experiment 1, I evolved controllers based on big maximum efficiency. The reason for this is that I wanted to obtain controllers which are capable of swimming

at high efficiency. Since I only considered positive efficiency to be valid, evolving controllers under this approach implicitly means evolving controllers with a larger efficiency range.

In Experiment 2, I evolved controllers based on big minimum efficiency. This approach implicitly forces all the measured efficiencies of the controller to be good because the GA is trying to pull up the worst efficiency each controller can achieve. Hence, it can be harder for the evolution system. However, the evolved controllers in this experiment should produce better results than those in Experiment 1, and a comparison of Table 6.2 and Table 6.3 shows that this is indeed correct.[7]

To determine pulse regularity, the condition $min\_fit\_oscil > 0.45$ is used. The threshold value of 0.45 is derived by Ijspeert based on experience (personal communication). Generally speaking, this value is good enough to distinguish neural waves which oscillate regularly from those which do not. It seemed to be suitable for my implementation as well. However, at the end of the evolutions, I realized that the GA had found a way to break this condition to pull up the efficiency. As the algorithm to compute the mechanical wavelength implicitly relies on pulse regularity (see below), relatively high efficiency values are obtained (refer to the discussion on "cheating controllers" in Subsection 6.5.1). Fortunately, the threshold problem appears only in two of the 60 evolved controllers.[8]

The fitness function that I proposed here rewards solutions which are good in all aspects. As the fitness factors are equally weighted, controllers with higher fitness may not necessarily be more efficient than those with lower fitness (compare the run5 and run20 controllers in Table 6.2). This problem can probably be corrected by including a weight on each fitness factor. However, this may bias the evolutions toward certain aspects of the overall required features. This might have happened to the evolutions in [Or et al., 2002]. The evolved robust controllers have low discrepancy values but the control of neural waves and the speed are not particularly good (i.e. they have very rough surfaces).

When plotting the speed vs. efficiency curves, I chose the data points manually.

[7]Note that the run12 controller in Experiment 1 achieved maximum efficiency of 0.86 by cheating. More on this in Subsection 6.5.1.

[8]These controllers are run12 (from Experiment 1) and run40 (from Experiment 2).

This can introduce bias. In some cases, the location of even a single data point can change the relation completely. The scatter plots are provided in Appendix D and E for those who want a more realistic picture of the relations.

The methods used to calculate the mechanical wavelength and efficiency have several limitations. According to Videler [1993], the measurement of kinematic parameters such as mechanical wavelength can be achieved accurately only as long as the mechanical wave crests propagating along the body are well pronounced and the amplitude is large even near the head. This should not pose a problem because these characteristics fit eel-like swimmers such as the lamprey. However, the two controllers with efficiency over 0.8 sometimes swim with a stiff body in approximately the first half of the body. This is similar to the sub-carangiform swimming mode described in Subsection 2.2.2. Under this situation, the measurement algorithm breaks down.

Finally, since time is finite in the simulations, there is an accuracy issue in calculations whenever time is involved. The error depends on the integration time step. The smaller the time step, the smaller the error (see the following derivations). In my experiments, a time step of 1 ms is used. To simplify the error calculation, I have made the following assumptions:

1. The directional vectors (at t_1 and t_2) used to calculate the speed are correct.
2. The traveling distance (C) is calculated correctly.
3. The lamprey swims in a fairly forward path. (i.e. not a lot of sudden change in swim direction at any time).
4. The locations of the lamprey at t_1 and t_2 are correct.

To calculate the mechanical wave speed (V), I computed the mechanical wavelength using the least square method. The constants a and b must be found to minimize the least square error:

$$\sum_{i=1}^{m}(y_i-(ax_i+b))^2$$

To miniminize this equation, differentiate it with respect to a and b to obtain the normal equations:

$$a\sum_{i=1}^{m} x_i^2 + b\sum_{i=1}^{m} x_i = \sum_{i=1}^{m} x_i y_i$$

and

$$a\sum_{i=1}^{m} x_i + b \cdot m = \sum_{i=1}^{m} y_i$$

The solution to the above system of equations is [Burden and Faires, 1993]:

$$a = \frac{m(\sum_{i=1}^{m} x_i y_i) - (\sum_{i=1}^{m} x_i)(\sum_{i=1}^{m} y_i)}{m(\sum_{i=1}^{m} x_i^2) - (\sum_{i=1}^{m} x_i)^2}$$

The error in mechanical wave speed (V) is therefore:

$$\pm\sum_{i=1}^{m} |\frac{da}{dx_i}\delta x_i|$$

where

$$\delta x_i = \pm 1ms$$

Once the error is found, it is converted to [m/s] and then expressed as a percentage of V (ErrorV).

To find the percentage error in swimming speed (ErrorU), I assume that the fastest achievable speed is $V = 0.8\ m/s$. When the lamprey is swimming along the x-axis, $V_x = V = 0.8\ m/s$. Since there is a time measurement error of $\pm 1\ m/s$, the maximum distance the lamprey can travel during this error period is:

$$\delta d = (0.8\ m/s) \cdot (0.001\ s) = 0.8\ mm$$

Assuming that the distance traveled by the lamprey is C, using the formula:

$$0.8 + 0.8 = x\% \cdot C$$

the percentage error in C, x, can be calculated. The reason for adding 0.8 twice is that there can be measurement error at both ends of the arc length C. ErrorU is therefore:

$$ErrorU = x\% + 2\%$$

Hence, the total percentage error in efficiency is:

$$ErrorE = ErrorU + ErrorV$$

Using data which correspond to the highest efficiency (1.073) I encountered when characterizing the swimming performance of the mechanical model using a sinusoidal controller (Subsection 5.4.5), $ErrorE = 2.81\%$.

The remaining errors are caused by the assumptions which I made in order to simplify the calculations. Thus, an error of $\pm 10\%$ in efficiency is acceptable based on the above analysis.

6.5 Discussion of results

Using the fitness function presented in this chapter, efficient swimming controllers have been evolved successfully. Most of them are more efficient than their corresponding prototypes. The neural configurations of the best individuals from the 20 evolutions are different even with the presence of the same prototype in the initial population. There is not much similarity in the way the segments are coupled.

Generally speaking, controllers based on the bigmin approach can achieve higher efficiency than those based on the bigmax approach.[9] Table 6.9 shows that all the evolved controllers have a maximum efficiency ≥ 0.58. Under the bigmax approach, three of the controllers have efficiency ≥ 0.65, the best of which has efficiency ≥ 0.7. When the bigmin approach is used, six of the evolved controllers have efficiency

[9]This observation holds even when I compare all 60 controllers.

≥ 0.65, three of which have efficiency ≥ 0.7. As the evolutions under the bigmin approach can produce more good solutions at the same time as those under the bigmax approach (they all started and terminated at the same time), it is better to evolve efficient controllers based on the bigmin approach.

Evolution approach	$e \geq 0.58$	$e \geq 0.60$	$e \geq 0.65$	$e \geq 0.7$
Bigmax (experiment 1)	10	9	3	1
Bigmin (experiment 2)	10	8	6	3

Table 6.9: Comparison of performance of the bigmax and bigmin approaches in terms of the efficiencies of the controllers. Since there is a cheating controller in each experiment, subtract one from each table element if they are considered to be invalid due to the breakdown of the mechanical wavelength calculation algorithm. Note that after these controllers are excluded, the number of controllers achieving efficiency ≥ 0.7 will be reduced to 0 and 2 for the bigmax and bigmin approach respectively. (More on cheating controllers in Subsection 6.5.1).

6.5.1 Discussion of the evolved controllers

In order to understand how the evolved controllers achieve high swimming efficiency, I have chosen the best five for further investigation. I looked at the characteristics of their neural waves as well as the corresponding swimming patterns. Based on these investigations, the controllers can be classified into two groups. The first group includes the run12 and run40 controllers (from bigmax and bigmin respectively) while the second group involves the run9, run20 and run29 controllers (all from bigmin). Controllers from the former group are called the "*cheating controllers*" as some of their neural waves contain irregular oscillations.

Figure 6.10 shows that the two controllers with the highest efficiency contain irregular neural waves.[10] Their *min_fit_oscil* values are 0.452 and 0.485 respectively (which is just over the threshold value of 0.45). The irregular neural waves cause the mechanical wave calculation algorithm to break down and return very short mechanical wavelengths (less than 0.1 m). This results in very high efficiency values. From

[10]The neural waves from some of the motoneurons shown in the top figure seem to have a period of approximately 300 msec.

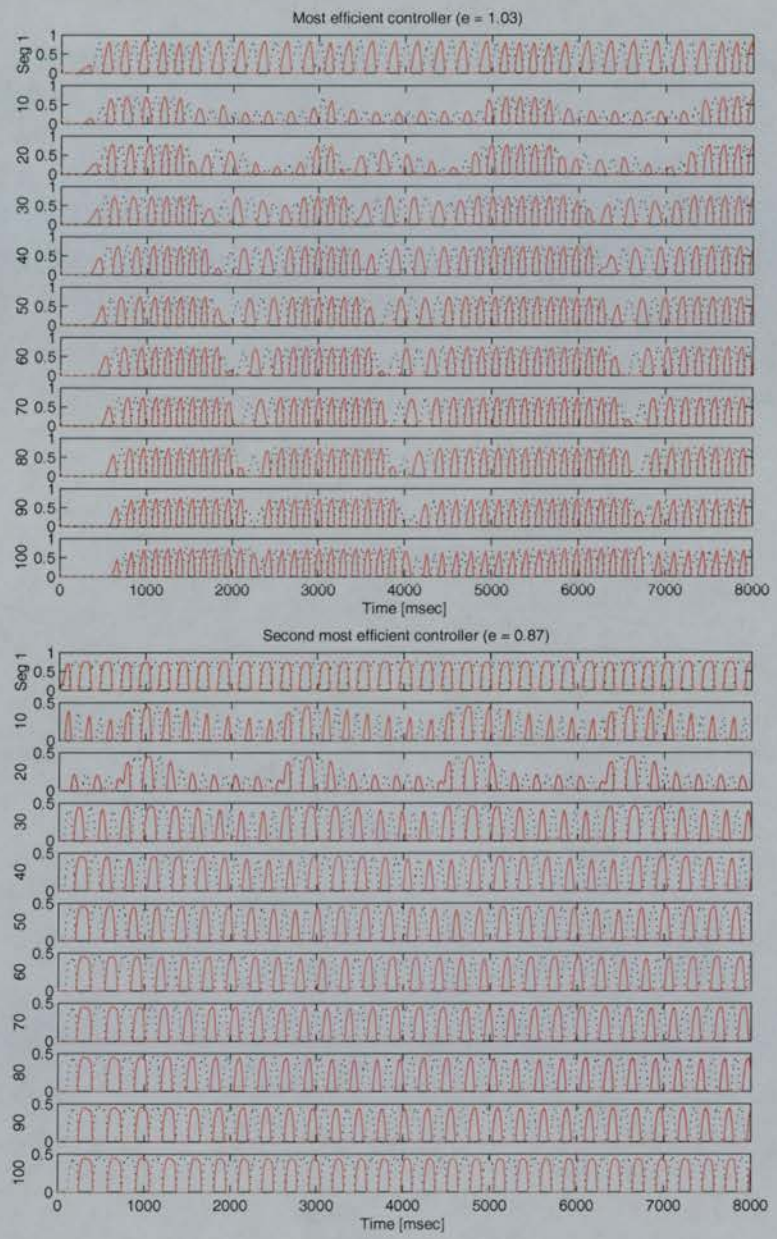


Figure 6.10: Irregular neural waves of the two most efficient controllers (by cheating). Solid lines represent outputs from the left motoneurons while dashed lines represent the outputs from the right motoneurons. Neural wave generated by the bigmin run40 controller (*top*). Neural wave generated by the bigmax run12 controller (*bottom*).

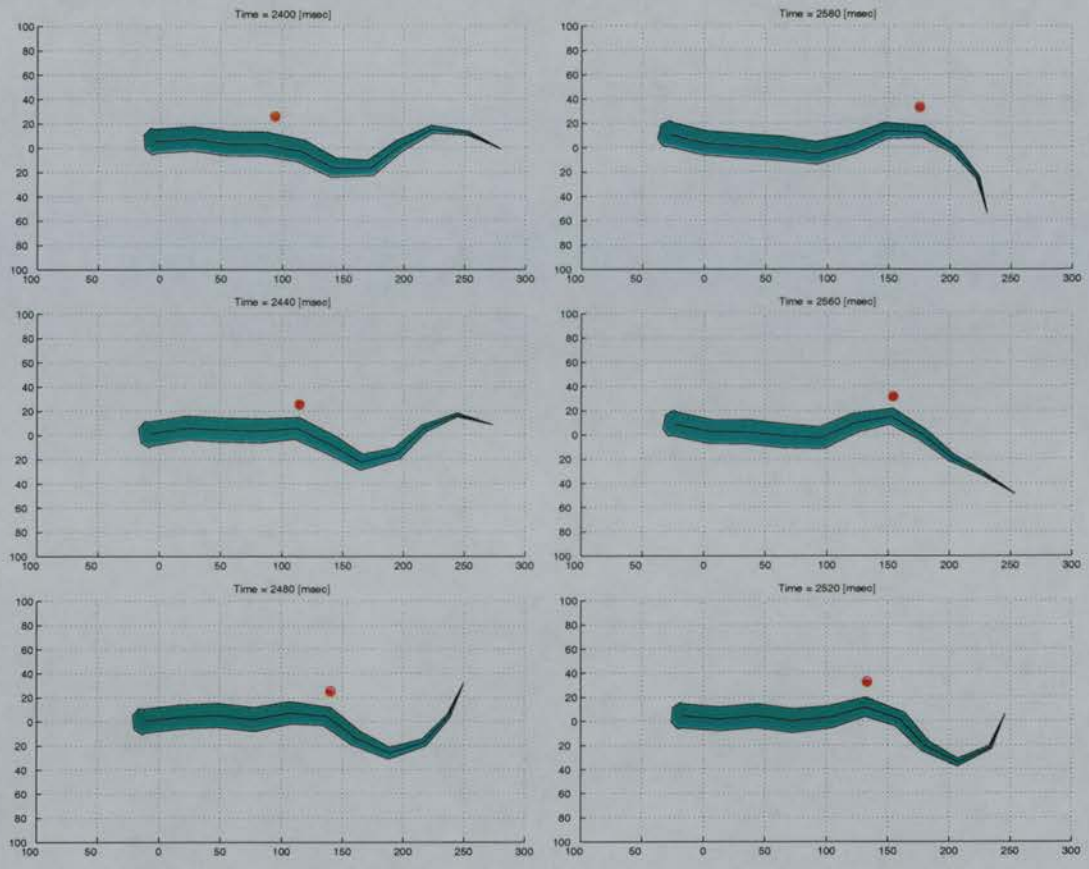


Figure 6.11: Sub-carangiform swimming of the lamprey. The snapshorts are 40 msec apart. The length and width of the swim-mill are in [mm]. From the figures (counter-clockwise from the top-left), a mechanical crest (indicated by a red dot) is traveling from the middle of the body to the tail in one swimming cycle.

computer animations (refer to Section 3.6), I have found that a lamprey embedded with either of these two controllers swims alternately between sub-carangiform and anguilliform swimming modes (see Figure 6.11). In either case, the lamprey swims with powerful tail movements. As for the remaining three controllers, the outputs from the motoneurons are regular. Figure 6.12 shows the neural wave of a typical controller (run9) from the second group.

The average amplitude and oscillation frequency of these three controllers are about 0.57 and 3.6 Hz respectively while the average swimming speed is about 0.34 m/s (refer to Table 6.10). These values are relatively low when compared with those

Figure 6.12: Neural wave of the third most efficient controller. Solid lines represent outputs from the left motoneurons while dashed lines represent the outputs from the right motoneurons. This controller (run9) was evolved under the bigmin approach. Note that this controller is later considered to be the most efficient, since when I explored the weakness of the mechanical wavelength calculation algorithm, the top two controllers were discovered to have been cheating.

of controllers evolved without taking efficiency into consideration. (Refer to p.90, Table 4.6, Chapter 4 in Ijspeert [1998] for example).[11]

As we require a controller which can both generate regular control signals and swim with high efficiency, the run9 controller evolved under the bigmin approach with controller 2 as its prototype is considered to be the most efficient one. The effect of excitations on efficiency for this controller is shown in Figure 6.13. The maximum efficiency achieved by this controller is 0.76, which is close to 0.8 achieved by the real lamprey [Williams, 1986, Carling et al., 1998].

[11]The average frequency and maximum speed for the evolved controllers reported in Ijspeert [1998] are 8.2 Hz and 0.54 m/s respectively.

Rank	(global, extra)	Amp	Freq [Hz]	Phase lag [%]	Mec λ [m]	Speed [m/s]	Wave speed [m/s]	Efficiency
1	(1.0, 40%)	0.76	5.56	1.22	0.04	0.34	0.33	1.03
2	(0.6, 140%)	0.45	3.60	1.69	0.02	0.27	0.31	0.86
3	(0.4, 100%)	0.54	3.52	1.60	0.12	**0.32**	0.42	**0.76**
4	(0.5, 150%)	0.60	4.03	1.29	0.13	0.37	0.55	0.68
5	(0.6, 30%)	0.58	3.23	1.61	0.15	0.34	0.49	0.68
Sin		0.54	3.52	1.60	0.16	**0.34**	0.58	**0.59**

Table 6.10: Comparison of neural and mechanical parameters for the five controllers with largest maximum efficiency. The controllers are run40, run12, run9, run29 and run20 (listed in order from top to bottom). Except for the run12 controller, these are evolved based on the bigmin approach. As the run40 and run12 controllers achieved high efficiency with irregular neural waves, the performance of the run9 controller is used to compare with that of a matched sinusoidal controller.

Although the efficiency is very high, the corresponding maximum swimming speed is about 0.32 m/s which is lower than the typical maximum speed of about 0.4 m/s. This means efficient energy utilization at the cost of speed, as observed in the real lamprey [Williams, 1986]. Since efficiency is the most important factor for survival and the lamprey can move from one place to another by attaching itself to a host, this swimming speed is acceptable from a biological viewpoint.

With regard to robustness in swimming speed against variations in body scales, although the bigmax run20 controller has the lowest discrepancy of about 0.1, its performance surfaces do not look smooth (see Figure B.9). This is similar to the observation made when analyzing the evolved robust controllers (reported in [Or et al., 2002]). It It may be that low discrepancy and good brainstem modulation are contradictory.

Note that some of the surfaces presented in Appendix B and C are not smooth. The reason for this is that although the fitness function rewards increase asymptotically in amplitude and frequency (with increase in global excitation) as well as increasing asymptotically in phase lag (with increase in extra excitation), the GA only looked at a small region of the excitation combination grids during fitness evaluation. This problem can be fixed by increasing the sampling resolutions. However, the time it takes to evaluate an individual controller will increase drastically.

From the speed vs. efficiency curves of all 20 controllers (see Appendix D and E),

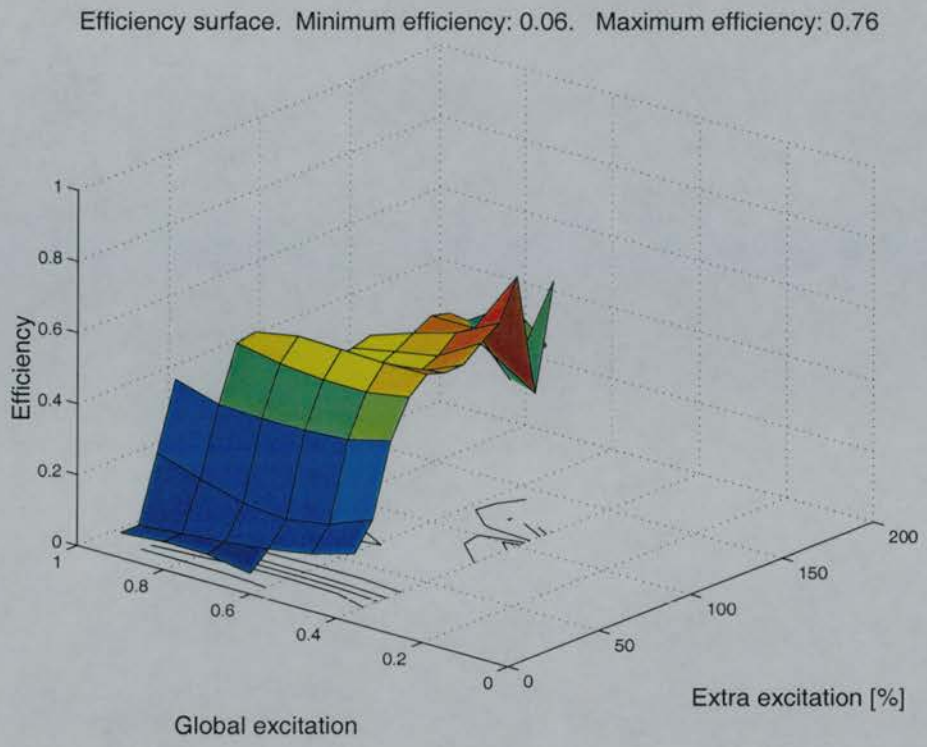


Figure 6.13: Efficiency surface for the official most efficient controller (bigmin run9). Note that efficiencies that cannot be measured (i.e. either $min\_fit\_oscil \leq 0.45$ or efficiency ≤ 0) are filtered. These filtered values correspond to the empty regions in the figure.

it seems that two types of controllers are evolved here. One has swimming efficiency increasing with speed (good from an evolution point of view [Blake, 1993]). The other type varies in efficiency in two phases: for speed below 0.3 m/s, efficiency increases with speed. After that, efficiency stays fairly constant. This is good from a robotics viewpoint because we want a controller which can exhibit a large range of speeds while being able to maintain high efficiency.

6.5.2 Comparison of the efficient controllers with a sinusoidal controller

Table 6.10 lists the neural parameters and swimming performance of the five controllers with highest efficiency with a matched sinusoidal controller. By matching, I

mean a sinusoidal controller whose amplitude, frequency and phase lag are tuned to match the corresponding parameters for the most efficient controller (the run9 controller). The table shows that the relatively high efficiency achieved by the first two controllers listed in the table is caused by the breakdown of the wavelength calculation algorithm (wavelength much less than 0.1 m). As the efficiency of these two controllers is invalid, the sinusoidal controller is compared with the run9 controller which is listed third in the table.

The comparison shows that although the analytic sinusoidal controller is able to achieve a slightly higher speed (0.34 m/s vs. 0.32 m/s), its efficiency is much lower than the run9 controller (0.59 compared with 0.76). Figure 6.14 shows the activity from the left motoneuron generated by the run9 controller with superimposed left motoneuron activity from the matched sinusoidal controller.

Although their neural parameters are matched, the pulse duration of the run9 controller is longer than that of the sinusoidal controller. Since the speed is inversely proportional to burst duration (Subsection 5.4.4), this agrees with the result that the swimming speed of the run9 controller is lower than that of the matched sinusoidal controller. Given that the sinusoidal controller can achieve a higher swimming speed, its efficiency would be greater if the mechanical wavelength of both controllers were the same (refer to Equations 6.2 and 6.3). The table shows that the mechanical wavelength of the run9 controller is shorter than that of the sinusoidal controller. Therefore, the run9 controller achieved a higher efficiency.

The reason why a lamprey driven by the run9 controller can have a shorter wavelength is due to the pulse shape. A comparison of the signals that control the first joint of the mechanical lamprey body shows that symmetrical pulses are generated by the sinusoidal controller. On the other hand, the pulses generated by the run9 controller are not symmetrical (Figure 6.14). Recall that the mechanical wavelength calculation algorithm requires the time instances at which a mechanical wave crest passes through different parts of the body. As the pulse shape can affect the amount of bending at each body link, the mechanical wavelengths of the two controllers can be different. It may be that the rather flat and unsymmetrical pulse shape of the run9 controller allows it to achieve a shorter mechanical wavelength.

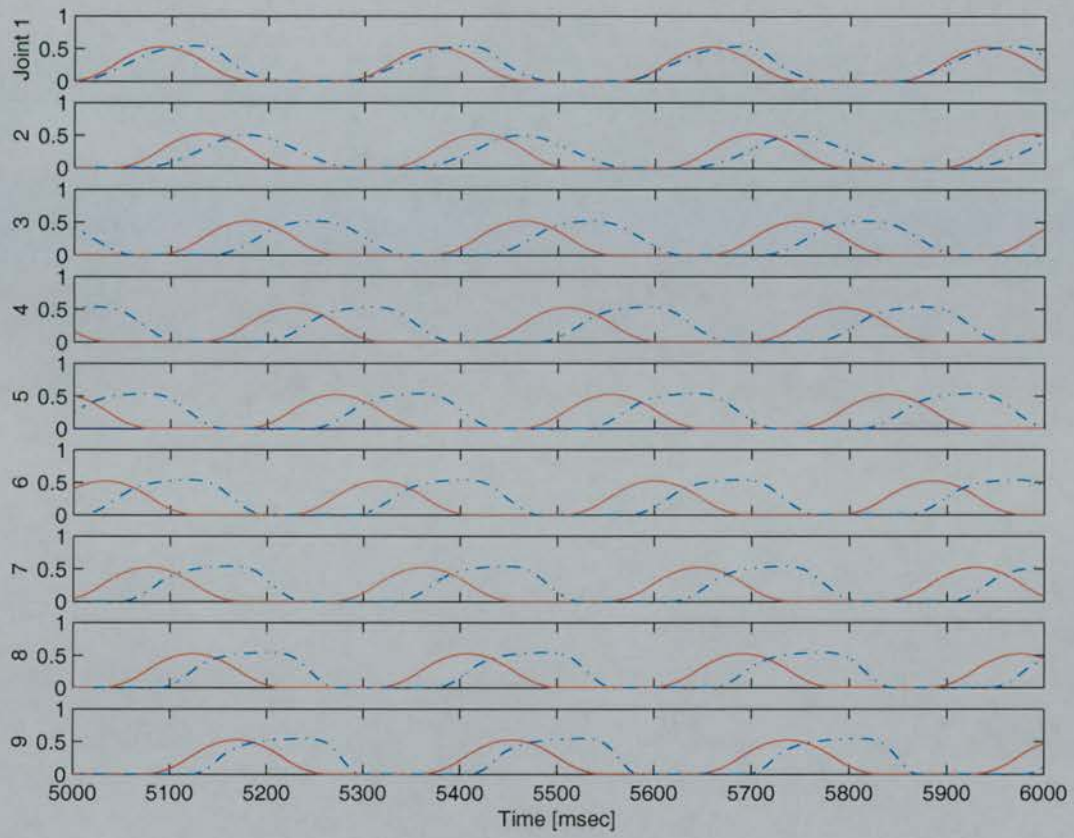


Figure 6.14: Averaged MNI activity from run9 controller with superimposed averaged MNI activity from a matched sinusoidal controller. The excitation combination used is (0.4, 100%). The amplitude is 0.54 while the frequency is 3.52 Hz. Phase lag is 1.60%. Dashed lines represent outputs from the run9 controller. Solid lines represent outputs from the matched sinusoidal controller. Note that the averaged neural wave from the run9 controller appears to have a larger phase lag than that of the matched sinusoidal controller. This illusion is caused by the reduction in neural connections at the ends of the CPG (and possibly by extra excitations at the first few segments). Although the phase lags of the signals controlling joints 3 to 8 match each other, those controlling joints 1, 2 and 9 do not. (Recall that phase lag is measured in the middle of the CPG.) Also, in my implementation, the neural waves from 10 segments are averaged to control one mechanical joint (see Subsection 2.5.2). The large phase lag of the original neural wave can slightly broaden the averaged pulse shape. However, the figure shows that there is actually no noticable difference in pulse duration between signals from the two controllers.

6.6 Summary

In this chapter, I successfully used GAs to evolve efficient swimming controllers. Most of the evolved controllers have a similar or even a better ability to maintain swimming speed despite variations in body scales than those evolved using a fitness function which takes into account the variation in performance with body scaling. Moreover, some of these controllers are robust in swimming efficiency against speed as well. It is interesting to note that the evolved efficient controllers have much lower frequency and speed than those evolved without taking swimming efficiency into consideration, as reported in Ijspeert [1998].

Based on experimental results, evolutions based on the bigmin approach produce more controllers with higher swimming efficiency than those based on the bigmax approach.

Most importantly, the GA has found a controller which can achieve the same efficiency as that observed in the real lamprey. The results reported here may lead to real biological experiments which in turn may provide a better understanding of the intersegmental couplings in real lamprey. So far, nobody knows in detail how the segments are connected to achieve high swimming efficiency. This remains the most difficult part of lamprey research. In the next chapter, the efficient controller (along with other controllers) will be investigated further with respect to optimality and robustness against noise in neural connections. The investigation leads to a list of features of intersegmental couplings which may potentially be responsible for high efficiency, optimality and robustness. Real biological experiments are required in order to confirm this hypothesis. Note that although the neural and mechanical models used in this thesis are simple, they are sufficient as tools for understanding biology via synthesis.

Chapter 7

Robustness of the swimming CPGs against random variation in neural connections

In Section 4.1, I discussed several types of robustness. In particular, I mentioned that it is very important for a robotic controller to be able to tolerate electronic imperfections (such as noise). The biological aspect is equally important. According to Ferrar et al. [1993], robustness against stochastic variation is an important property for biological systems. A controller which can achieve the highest swimming speed and/or efficiency but is unable to tolerate random noise in the networks is undesirable in the real world. As maximum swimming speed and efficiency are important to the survival of the lamprey, I investigate how these two variables are affected by controllers with random variations in neural connections. Given that each connection strength in the CPG is calculated from the number of extensions to neurons in both rostral and caudal segments, this is analogous to subjecting the control circuitry to noise. Two sets of experiments are conducted in this chapter. The first set consists of four experiments which investigate how varying the segmental and/or intersegmental connections affects the maximum swimming speed and efficiency. The second set of experiments investigates how variations in both segmental and intersegmental connections affect swimming speed across a wide range of global and extra excitations. As part of the

analysis, I compare the optimization of the controllers. Through comparative analysis, a list of features in intersegmental couplings which might be responsible for the optimization, robustness and efficiency of the swimming CPG is proposed. The four controllers under investigation are:

1. *The Biological Controller*, hand-crafted by Ekeberg [1993] and based on physiological data.
2. *Controller 2*, evolved by Ijspeert et al. [1999] using Ekeberg's segmental oscillator and intersegmental couplings evolved by GA.
3. *The Hybrid Robust Controller*, obtained through the evolution described in [Or et al., 2002].
4. *The Efficient Controller*, the run9 controller obtained through evolution using the bigmin approach described in Section 6.1.

7.1 Investigation of the effect of varying the segmental and/or intersegmental connections on the maximum swimming speed and efficiency

7.1.1 Description of experiments

To test the robustness of the controllers under random variations in connection weights, I conducted the following experiments on each of them, and then compared their performance in terms of either maximum achievable speed or efficiency.[1] For a controller to be robust, its maximum achievable speed or efficiency should be similar to the speed or efficiency obtained when noise is not present.[2]

[1]For reason why the maximum speed and efficiency are used as evaluation criteria, refer to Chapter 4 and Chapter 6 respectively.

[2]Although the term 'noise' generally refers to random variation over time, here I use it to mean random variation in neural connections only at the beginning of the simulations.

- *Experiment 1: Robustness of swimming speed against random variation in segmental connection weights.* For each controller, the excitation combination which corresponds to the maximum swimming speed is chosen as the brainstem input. The connection weights between neurons at a prototype segment are subjected to random variation (with uniform distribution) in weight from $\pm 5\%$ to $\pm 25\%$ in steps of 5%. One hundred copies of this segmental oscillator are then combined to form the swimming CPG (with the number of extensions to neighbouring segments unchanged). Note that at each amount of variation, 100 runs (each of which has different seeds for the random number generator) of neural and mechanical simulations are performed to obtain the average maximum speed for all 100. Thus, for each controller, five averaged speeds are computed across the amount of variation under investigation.

- *Experiment 2: Robustness of swimming speed against random variation in segmental connection weights and intersegmental couplings.* This is the same as Experiment 1 but in addition the numbers of rostral and caudal extensions are subjected to random variation. Each extension can be varied by up to three segments.

- *Experiment 3: Robustness of swimming efficiency against random variation in segmental connection weights.* This is the same as Experiment 1 but the excitation combination which corresponds to the maximum swimming efficiency is chosen as the brainstem input instead of maximum swimming speed. At each amount of variation, 100 runs of neural and mechanical simulations are performed to obtain an average of all 100 maximum efficiency values. Thus, for each controller, five averaged efficiency values are computed across the noise range under investigation.

- *Experiment 4: Robustness of swimming efficiency against random variation in segmental connection weights and intersegmental couplings.* This is the same as Experiment 3 but in addition the number of rostral and caudal extensions are subjected to random variations.

Note that as each calibrated connection strength is based on the total number of connections to neurons of neighboring segments, the same controllers are subjected to more variations in Experiments 2 and 4 than in Experiments 1 and 3.

7.1.2 Methods

Each of the experiments was conducted with the following stages:

- *Stage 1: Random variation of connection weights.*
  After the chromosomes which encode the segmental and intersegmental connections for a controller are loaded, the connection matrices (one for segmental connection weights and the others for rostral and caudal extensions) are formed. Depending on the experiment, either the former matrix alone or all three matrices are subjected to random variation. Note that random variation is only applied at the beginning of each simulation. To avoid possible turning and to perserve consistency with experiments performed in other chapters, left-right symmetry is maintained.

- *Stage 2: Neural simulations of the swimming CPG.*
  The excitation combination corresponding to maximum swimming speed or efficiency is chosen as the brainstem input. Neural simulations are performed. The neural parameters (amplitude, frequency and phase lag) of the outputs from the simulated swimming CPG are recorded.

- *Stage 3: Mechanical simulation of the lamprey swimming in a 2D environment.*
  At this stage, mechanical simulations are performed with the neural waves obtained in Stage 2. The swimming speeds and efficiencies are recorded.

- *Stage 4: Percentage change in speed/efficiency calculation.*
  Once the averaged speed/efficiency at each amount of variation is calculated for each controller, the corresponding percentage change in speed/efficiency, with respect to the original ones (without noise), is calculated.

- *Stage 5: Comparative analysis of swimming controllers within the same experiment.*

As mentioned, after the connection matrices are formed, random noise is added to the neural networks. Each connection weight is modified using:

$$new\_weight = original\_weight + original\_weight \cdot rand \cdot noise\_range$$

where *rand* is a small random number evenly distributed within [-0.5, 0.5] and the noise range can be any value between 0.1 and 0.5 in steps of 0.1. This corresponds to randomly varying the connection weight by ± 5 to $\pm 25\%$ (in steps of 5%). Using a similar equation, each intersegmental extension can be varied by up to ± 3 segments.[3] Table 7.1 shows how the noise range is related to the maximum amount of variation in connection weight and intersegmental extensions. Note that as 100 copies of the same segmental oscillator is used to form the entire swimming CPG, all segments were subjected to the same amount of noise. In other words, the extensions from each source neuron to the corresponding target neuron in the rostral/caudal neighbours are the same regardless of its location along the CPG.

Noise range [%]	Maximum range of possible variation in connection weights [%] and intersegmental extensions
10	± 5 [0]
20	± 10 [± 1]
30	± 15 [± 1]
40	± 20 [± 2]
50	± 25 [± 3]

Table 7.1: Conversion table from the noise range to the maximum ranges of possible variation in connection weights and intersegmental extensions (in brackets).

7.1.3 Results and Discussion

Figure 7.1 shows how the speeds of the four controllers are affected at each given amount of random variation (noise range) while Figure 7.2 shows how the efficiency

[3]Since the weights are in real numbers, each calculated number of extensions is converted into an integer value by truncating the digits after the decimal point. The result is then assigned to either the rostral or caudal extension matrix.

of the controllers is affected. For comparison purposes, the maximum speed and efficiency values achieved by the controllers when there is no noise are shown in Table 7.2.

Note that as I am interested in how much damage/gain a particular controller gets when random changes are made relative to its original form, I measure the relative change in performance. Hence, percentages rather than absolute values are used. A bigger percentage drop in speed (efficiency) means that the controller's original neural configuration is better optimized (i.e. at the peak of a hill) for high speed (efficiency) than that of other controllers. Note that if the absolute speed (efficiency) of this controller is still better than the rest of the controllers, it means that its peak is relatively higher.

Controller	Speed [m/s] at excitation combination	Efficiency at excitation combination
Biological	0.45 (0.6, 200%)	0.58 (0.3, 150%)
Controller 2	0.49 (0.9, 40%)	0.59 (0.4, 50%)
Hybrid Robust	0.49 (0.9, 0%)	0.69 (0.6, 80%)
Efficient	0.51 (0.9, 40%)	0.76 (0.4, 100%)

Table 7.2: Maximum speed and efficiency achieved by different controllers without noise in the CPGs. The excitation combination at which the maximum speeds and efficiencies occur are give in parenthesis. The first number represents global excitation while the second number represents extra excitation from the brainstem.

Experiment 1: Robustness of swimming speed against random variation in segmental connection weights.

The top graph in Figure 7.1 shows how varying the segmental connection weights affects the swimming speed. For all controllers it appears that increasing the variation in weight tends to decrease the swimming speed. This tendency implies the possibility that the neural configurations of the four controllers are already nearly optimized for high speed swimming. The speed of controller 2 changes the least: with $\pm 25\%$ variation in segmental connection weights, its speed only drops by 13.3%. As for the biological controller and efficient controller, the maximum drop in their speed is 34.7% and 26.3% respectively. The hybrid robust controller performs the worst. Even with

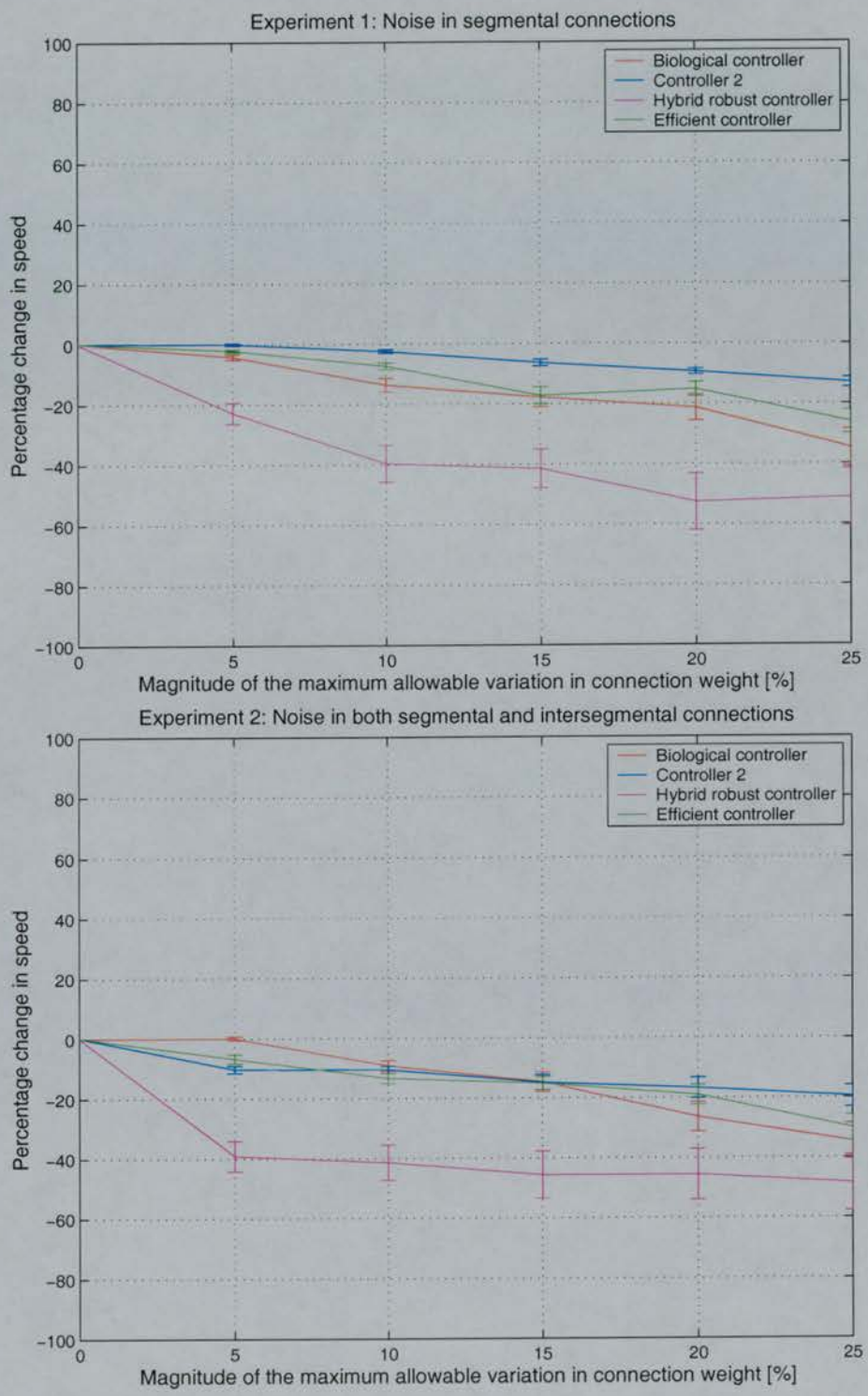


Figure 7.1: Percentage change in the averaged speed, with respect to maximum speed obtained with the same CPGs but without noise. The effect of noise in segmental connection weight (*top*). The effect of noise in both segmental connection weights and intersegmental extensions (*bottom*). Note that each averaged speed is the average of 100 runs with different random seeds. The error bars correspond to standard errors.

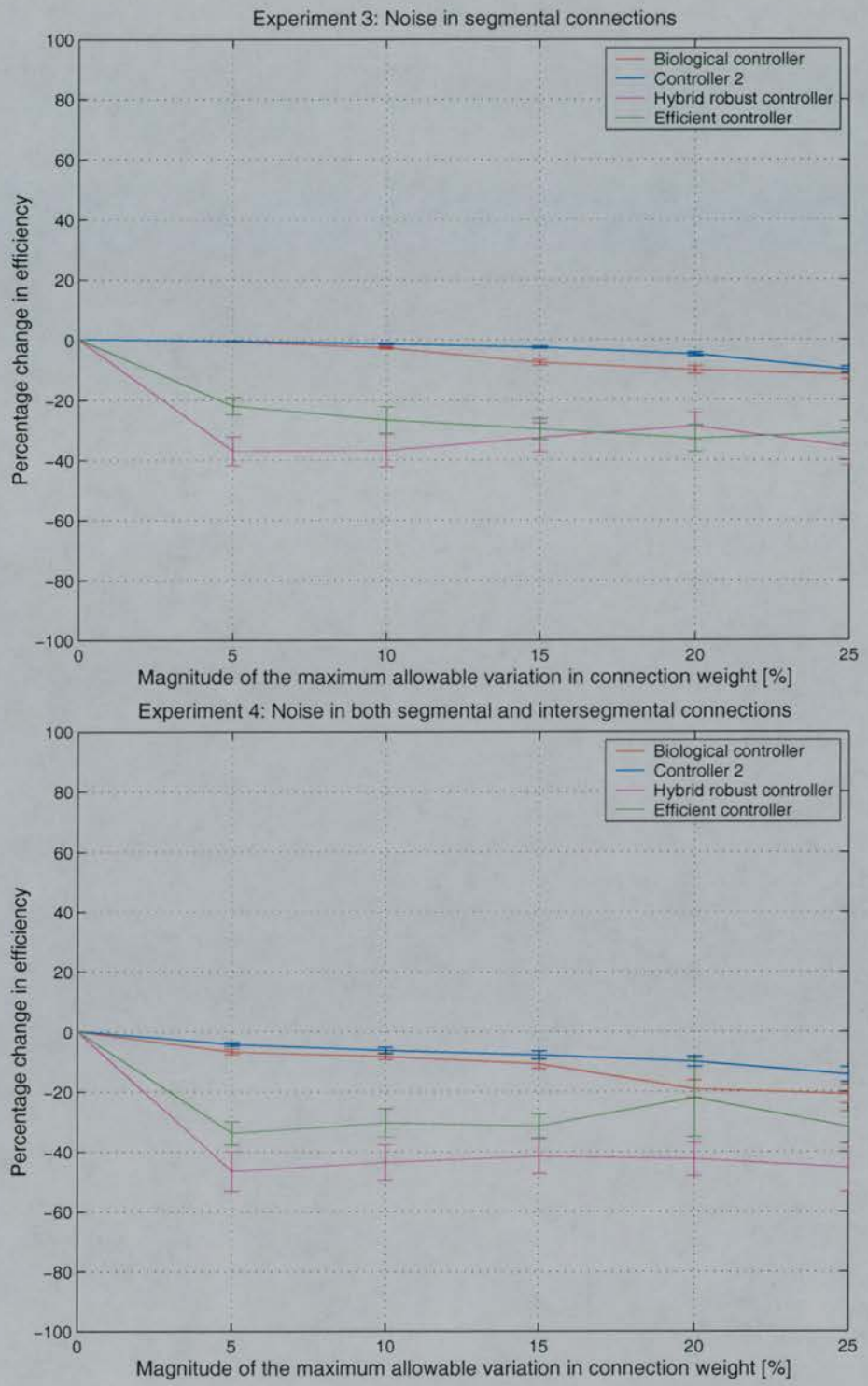


Figure 7.2: Percentage change in the averaged efficiency, with respect to maximum efficiency obtained with the same CPGs but without noise. The effect of noise in segmental connection weights (*top*). The effect of noise in both segmental connection weights and intersegmental extensions (*bottom*). Note that each averaged efficiency is the average of 100 runs with different random seeds. The error bars correspond to standard errors.

only ±5% variation in weight, its swimming speed drops by 22.8% (which is worse than the worst case of controller 2). As the amount of variation increases, the speed of the hybrid robust controller continues to drop. At ±25% variation in segmental connection weights, the speed drops by more than a half. The percentage change in the speed curve for this controller indicates that noise in the segmental connection weights alone has a big negative impact on the swimming speed.

To summarize, in terms of robustness in speed against noise in segmental connections, controller 2 is the most robust. It is followed by the efficient controller, the biological controller and then the hybrid robust controller. As the biological controller and the hybrid robust controller are the least robust, their neural configurations are more optimized for high speed swimming than the other two controllers. The percentage speed change of each controller, with respect to the maximum speed obtained when there is no noise in the segmental connection weights reported in Table 7.2, is shown in Table 7.3.

	Range of variation in segmental connection weight				
Controller	±5%	±10%	±15%	±20%	±25%
Biological	-4.2	-13.6	-17.8	-21.4	-34.7
Controller 2	-0.2	-2.7	-6.5	-9.6	-13.3
Hybrid robust	-22.8	-39.6	-41.5	-52.6	-51.2
Efficient	-2.4	-7.5	-17.4	-15.4	-26.3

Table 7.3: Percentage change in speed for all controllers in Experiment 1. Each percentage represents the maximum range of possible variation in segmental connection weights.

Experiment 2: Robustness of swimming speed against random variation in segmental connection weights and intersegmental couplings.

The bottom graph in Figure 7.1 shows how varying the segmental connection weights and intersegmental couplings affects the swimming speed. For all controllers, swimming speed tends to decrease with the presence of noise. Note that at ±5% noise in both segmental and intersegmental connections, there is no change in speed for the biological controller. As the variation continues to increase, the speed of this controller

drops monotonically. The swimming speed achieved by the rest of the controllers drops immediately with the presence of noise. Controller 2 has its speed drop by less than one fifth in the worst case, which is reasonably good. One interesting point to note is that, like in Experiment 1, the speed of this controller stays fairly constant across different noise ranges. Hence, controller 2 is the most robust in swimming speed against stochastic variation in any kind of neuronal connection. This may be due to its unique neural organization, which allows more flexibility than other controllers.[4] As for the hybrid robust controller, it has the greatest reduction in speed. Even with $\pm 5\%$ variation, its swimming speed drops by about 40%, which is worse than the worst case of any other controller except itself. Except for the 30% drop in speed at $\pm 25\%$, the efficient controller behaves similarly to controller 2.

The percentage speed change of each controller, with respect to the maximum speeds reported in Table 7.2 (obtained when there is no noise in the CPGs) is shown in Table 7.4.

	Range of variation in both segmental and intersegmental connections				
Controller	$\pm 5\%$	$\pm 10\%$	$\pm 15\%$	$\pm 20\%$	$\pm 25\%$
Biological	0.0	-9.1	-14.7	-26.5	-34.7
Controller 2	-10.4	-10.6	-14.9	-16.9	-19.8
Hybrid robust	-39.2	-41.5	-45.7	-45.7	-48.6
Efficient	-6.9	-13.5	-15.2	-19.2	-30.3

Table 7.4: Percentage change in speed for all controllers in Experiment 2. Each percentage represents the maximum range of possible variation in segmental connection weights and intersegmental couplings.

Experiment 3: Robustness of swimming efficiency against random variation in segmental connection weights.

The top graph in Figure 7.2 shows that the efficiencies of the biological controller and especially controller 2 are extremely stable across the variation under investigation.

[4] A comparison of the neural organizations of the biological controller and controller 2 is given in Section 5.3.

Although the percentage drop in efficiency increases with the amount of variation in weight, the absolute efficiency at each noise range for these two controllers is similar to that of the corresponding originals. To be specific, the efficiency of the biological controller only drops to a maximum of about 12% in the worst case while that of controller 2 drops to about 10% at most. The extremely good performance of these two controllers indicates that their efficiencies are less affected by noise in the segmental connections. This is good for the lamprey because robustness in efficiency is more important than robustness in swimming speed. As for the hybrid robust controller and the efficient controller, their tolerance against noise is relatively poor. With only $\pm 5\%$ variation in segmental connection weights, their efficiencies drop by about 37% and 22% respectively. This is worse than the worst case of both the other two controllers. However, as the amount of noise continues to increase, the percentage change in efficiency of these two controllers stays fairly robust (with about 10% change at the extremes of the noise range). Note that the percentage drop in efficiency for the hybrid robust controller generally decreases as the amount of variation increases. From the results of Experiment 1, we know that the speed for this controller drops as the amount of variation in segmental connections increases. Since efficiency is defined as the ratio of U to V, this implies that the mechanical wave speed (V) drops faster than the swimming speed (U). Given that the mechanical wave speed is defined as the ratio of mechanical wavelength to the mechanical period and that this period is fixed as the neural wave remains the same, the mechanical wavelength must be dropping rapidly with the increase in noise.

The percentage efficiency change of each controller, with respect to the maximum efficiency reported in Table 7.2 (obtained when there is no noise in the segmental connection weights) is shown in Table 7.5.

Experiment 4: Robustness of swimming efficiency against random variation in segmental connection weights and intersegmental couplings.

The bottom graph in Figure 7.2 shows that both the biological controller and controller 2 are fairly robust in efficiency against the presence of noise in both segmental and intersegmental connections. However, their absolute efficiencies are worse than

	Range of variation in segmental connection weights				
Controller	±5%	±10%	±15%	±20%	±25%
Biological	-0.7	-2.7	-7.6	-10.1	-11.7
Controller 2	-0.7	-1.5	-2.7	-5.0	-10.2
Hybrid robust	-36.9	-36.8	-32.5	-28.7	-35.8
Efficient	-22.2	-26.8	-29.9	-33.9	-31.2

Table 7.5: Percentage change in efficiency for all controllers in Experiment 3. Each percentage represents the maximum range of possible variation in segmental connection weights.

they were in Experiment 3. As the amount of variation increases, controller 2 becomes more robust than the biological controller. The percentage drop in efficiency for these controllers at a maximum possible change of ±25% in segmental connection weights and intersegmental couplings is at most 15% and 21% respectively. Given that these percentage drops are relatively low, they are acceptable. As in Experiment 3, the hybrid robust controller and the efficient controller behave similarly to each other. When compared with the biological controller and controller 2, their absolute performances are much worse. Even at only ±5% variation, their efficiencies drop to 46.6% and 33.9% respectively. The performance of the hybrid robust controller is the worst overall. Its drop in efficiency at each noise range is almost half of the original. Note that after the initial drop at ±5% variation in any neural connection, the efficiencies of both the hybrid robust controller and the efficient controller stay fairly constant. The variation in efficiency is less than 5% for the hybrid robust controller and about 3% for the efficient controller (except at ±20% variation in connections).

The percentage efficiency change of each controller with respect to the maximum efficiency reported in Table 7.2 (obtained when there is no noise in the CPGs) is shown in Table 7.6.

7.1.4 Behavior of controllers across the four experiments

Except for the efficient controller, each controller performs similarly across the different experiments. In general, the four controllers under investigation fall into three categories. In the first case, the absolute speed/efficiency of the biological controller

	Range of variation in both segmental and intersegmental connection weights				
Controller	±5%	±10%	±15%	±20%	±25%
Biological	-6.7	-8.2	-10.7	-19.1	-20.8
Controller 2	-4.4	-6.4	-7.9	-10.1	-14.4
Hybrid robust	-46.6	-43.6	-41.6	-42.4	-45.3
Efficient	-33.9	-30.5	-31.6	-22.1	-31.8

Table 7.6: Percentage change in efficiency for all controllers in Experiment 4. Each percentage represents the maximum range of possible variation in segmental connection weights and intersegmental couplings.

and controller 2 at each amount of variation (noise range) is close to that of the originals. It means that these controllers are at the top of the plateau in the fitness landscape. However, the hybrid robust controller forms a class of its own, as its absolute speed/efficiency drops significantly even when there is only a small variation of $\pm 5\%$ in any kind of neural connection. In other words, this controller is more at the peak in the fitness landscape. The third case is the efficient controller which behaves differently depending on the experiment. In terms of robustness in swimming speed (Experiments 1 and 2), it behaves similarly to both the biological controller and controller 2, but its behavior in terms of robustness in swimming efficiency (Experiments 3 and 4) is somewhere between the two other classes of controllers.

Controller 2 performs best in all the experiments. Its swimming speed and efficiency stay relatively constant across the different amounts of variation under investigation. Thus, it is the most robust in both speed and efficiency against noise in the networks.

Although the hybrid robust controller is the most robust in maximum swimming speed against changes in body scales,[5] its performance is the worst in all the experiments in this chapter. Thus, this controller is the most unstable when noise is present in the network system.

Having considered the neural characteristics (amplitude, frequency and phase lag) and swimming performance of each controller across different experiments and dif-

[5] In Or et al. [2002], I considered a controller with the lowest discrepancy value to be the most robust.

ferent amounts of variation (see Figures 7.3 and 7.4), I have observed that there is a tendency for the amplitude of the motoneuron outputs from each controller in each experiment to be least affected by the random seed (i.e. each set of 100 runs gives similar amplitudes). This is followed by the frequency of oscillations, phase lag and swimming speed/efficiency. It could be that since the measurements of these parameters depend on an increasing number of other factors, the variations add up. Hence, even with the same amount of variation, the amplitude is less affected when compared with the others. Also, as the amount of variation in connection weights increases, the standard error in both neural characteristics and swimming performance tends to increase as well.

7.2 Investigation of how varying the segmental and intersegmental connections affects swimming speed at different excitation combinations

In this section, I investigate robustness in swimming speed against noise in both segmental and intersegmental connections at different excitation combinations. The reason for conducting this investigation is to find out if the controllers are robust across a wide range of global and extra excitations. A controller which is extremely robust but only at a very limited excitation region is not useful for practical purposes.

7.2.1 Methods

The procedures used in Experiment 2 of the previous investigation (Section 7.1) and those for performing the set of neuromechanical simulations described in Section 3.6 are used here with the following modifications:

- Once the controller under investigation is loaded into the program, each segmental and intersegmental neural connection is subjected to noise before the set of neuromechanical simulations is performed. In this investigation, the noise range can be 0.1, 0.2 or 0.4, which corresponds to randomly varying any connection

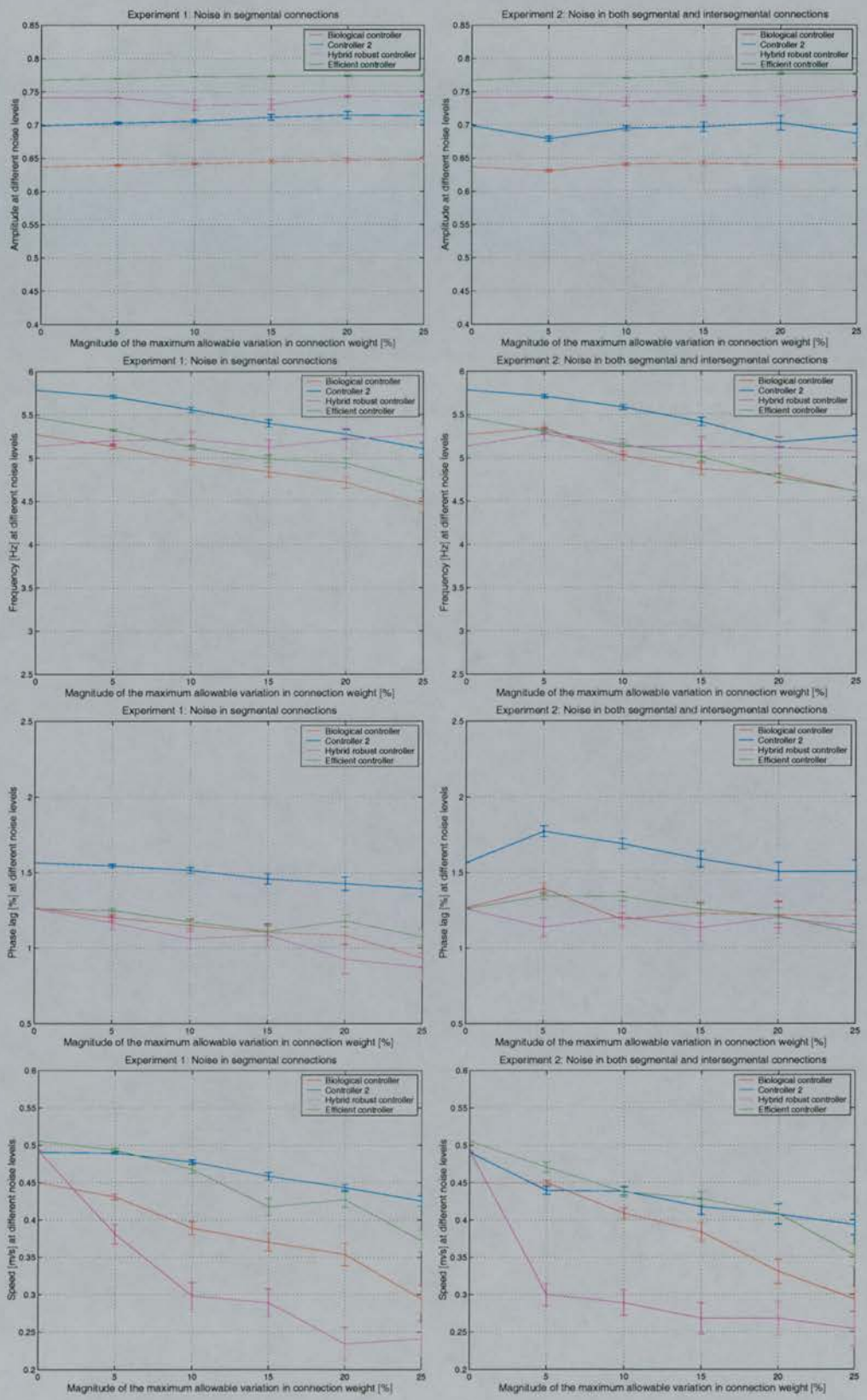


Figure 7.3: Changes in neural characteristics and speed caused by noise in Experiments 1 (*left*) and 2 (*right*). The error bars correspond to standard errors.

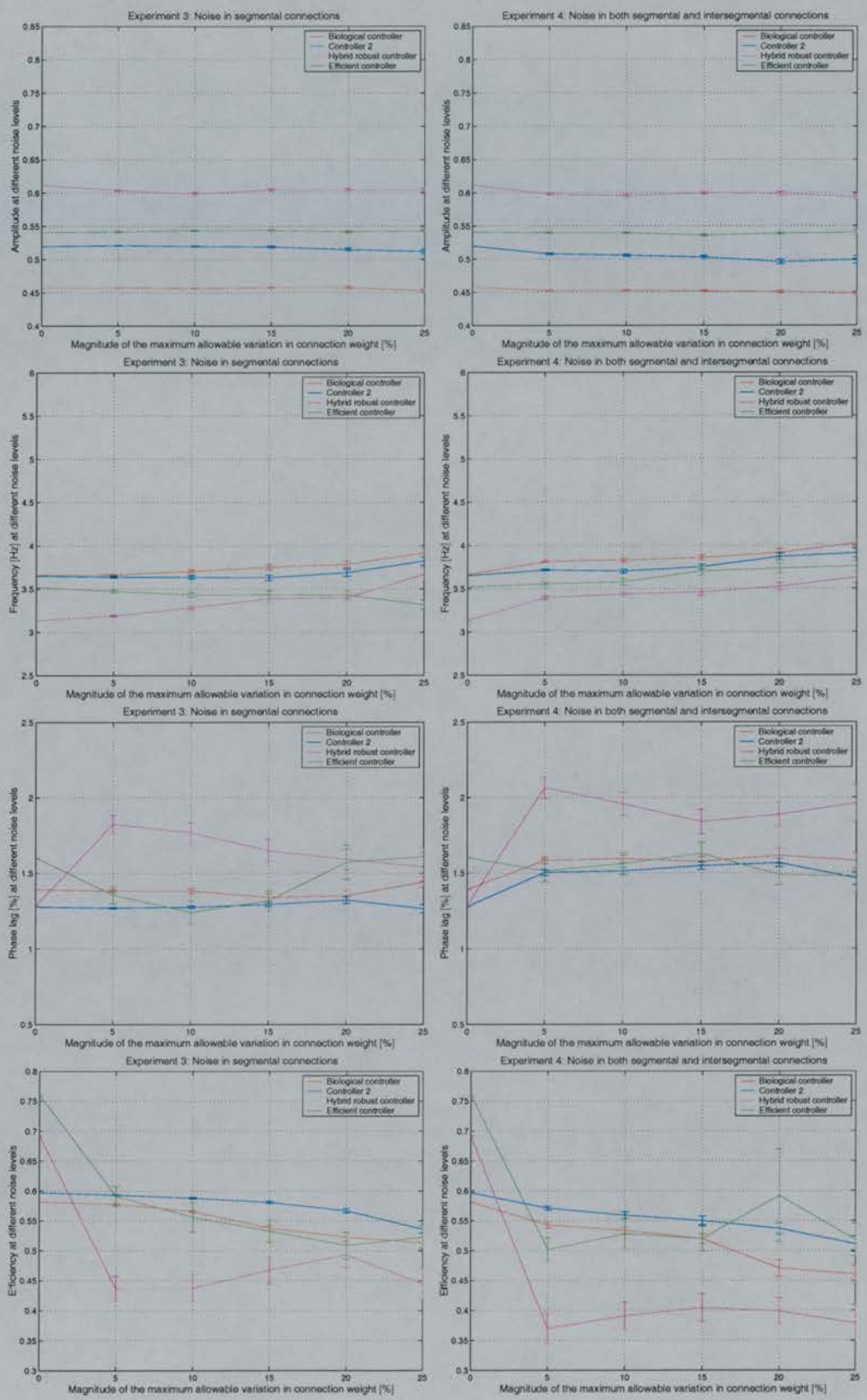


Figure 7.4: Changes in neural characteristics and efficiency caused by noise in Experiments 3 (*left*) and 4 (*right*). The error bars correspond to standard errors.

weight by $\pm 5\%$, $\pm 10\%$ or $\pm 20\%$ respectively. Each of the intersegmental extensions can be varied by 0, ± 1 or ± 2 segments.

- Since I am interested in the overall behavior of the controllers, a coarse sampling is taken. The global excitation ranges from 0.2 to 1.0 in steps of 0.1, while extra excitation ranges from 0% to 200% in steps of 20%.

For each controller and noise range, the above procedure is repeated 25 times each with a different random seed. This amounts to a total of 300 speed surfaces (data files) available for analysis.[6]

7.2.2 Description of evaluation criteria

In order to compare the controllers' performance at different noise ranges and excitation combinations, three sets of programs are written to evaluate the controllers in different ways. This corresponds to Analyses 1, 2 and 3. The first analysis investigates each controller's ability to produce forward swimming. The second analysis investigates whether the controllers are optimized for high speed swimming. The final analysis compares the robustness of the controllers using an evaluation criterion different from the one used in Section 7.1.

- *Analysis 1: Ability to produce forward swimming*

  For each controller and noise range under investigation, the following procedure is used:

  1. For purposes of comparison, load the original speed matrix and call it *original_grid*. By "original speed matrix", I mean the matrix which stores the speeds achieved by the controller at different excitation combinations (without the presence of noise in neural connections). Note that this matrix

[6] Four controllers, 25 runs at three different noise ranges.

was created in previous studies. As coarse sampling is considered here, the size of the matrix is reduced to 11 x 9.[7]

2. Create three matrices of the same size as *original_grid*. Call them *result_Good*, *result_Bad* and *result_Neutral*. Initialize each matrix element to zero.

3. For each of the 25 runs at each excitation combination, if the new speed is working but the old speed was not, add one to the *result_Good* matrix. On the other hand, if the new speed is not working but the old one was working, add one to the *result_Bad* matrix. Otherwise, do nothing (the neutral case). Note that in the discussion that follows, "new speed" means speed achieved by the controller with the presence of noise in neural connections, and "old speed" means the speed achieved by the controller without the presence of noise. In this analysis, I consider speeds above zero to be "working" or "success",[8] since in this dissertation, I focus on forward swimming. Being able to distinguish "forward" from "non-forward" swimmings allows me to measure the frequency at which the controllers break down (i.e. are not swimming forward).

4. Compute each element in the *result_Neutral* matrix by subtracting 25 from the corresponding element in the *result_Good* and *result_Bad* matrices.

5. Since I am interested in the proportion of successes and failures at each excitation combination, each element in all three matrices is divided by 25. The results are plotted using a stacked bar graph, so that, in other words, there is a stacked bar graph for each excitation combination under investigation.

At the end of the analysis, 16 grid matrices (four controllers with performance at 0%, $\pm$5%, $\pm$10%, and $\pm$20% variation in connection weights) are created for

[7]Recall that I used a finer resolution of global and extra excitations in previous studies. As a result, rows and columns that are not considered here are removed from the original matrices.

[8]Note that under this definition, extremely small swimming speed (such as 0.000001 m/s) is still considered to be "working".

comparison purposes.

- *Analysis 2: Optimization of the neural configuration*

  In order to investigate whether the controllers are optimized to achieve the highest swimming speed at different excitation combinations, the procedure described in Analysis 1 is repeated with the following change in Step 3:

  If the new speed is higher than the old speed, add one to the *result_Good* matrix. On the other hand, if the new speed is lower than the old speed, add one to the *result_BAD* matrix. Otherwise, do nothing (the neutral case).

- *Analysis 3: Robustness in swimming speed*

  In order to compare the controllers in terms of speed change due to the variation in neural connections, the average of how much each new speed is changed with respect to the old speed is calculated using the following formula:

$$variability = \frac{\sum_{i=1}^{25} \left| \frac{new\_speed_i - original\_speed}{original\_speed} \right|}{25}$$

  Note that if the original speed is zero (i.e. speed cannot be measured), variability is undefined. At each excitation combination, small variability means that the controller is generally able to maintain similar speeds (among the 25 runs) regardless of random variation in neural connections. Hence, the controller is robust. On the other hand, if the speed changes a lot (or when the controller switches from forward to backward swimming or vice versa), variability becomes large. To visualize the controllers' robustness at different noise ranges and excitation combinations, a bar graph in red of length proportional to variability is plotted at each excitation combination. Impressionistically, it appears to be the case that variability less than 1 leads to few changes in speed (among

the 25 runs) despite the presence of noise, whereas variability over 1 corresponds to lots of speed change. In order to clearly distinguish the two cases, the length of each bar graph is standardized to 2. This means that an excitation grid which is completely filled with red color represents poor robustness.[9]

7.2.3 Results and Discussion

Analysis 1: Ability to produce forward swimming

The results from the first analysis are shown in Figure 7.5. (For a zoom-in of the figures, refer to Appendix F.)

Figure 7.5 shows that each controller has a different capacity to cope with noise in neural connections. For each controller, there are boundary regions. These regions mark the areas of excitation combinations near which the original controller fails. Such regions are the most sensitive to noise since increasing the amount of variation in connections can affect both the number of successes and the number of failures. In other words, increasing the amount of variation not only increases the chance of failure but also increases the chance of success. This phenomenon can be explained as follows: the excitation combinations which make up the boundary located at higher global excitations correspond to inputs from the brainstem at which the controller can operate. When the neural configuration is varied, the chance that these excitations cause the modified networks to saturate becomes high. On the other hand, the excitation combinations in the original configuration which could not lead to forward swimming (due to saturation) have a chance to produce forward swimming with each modified neural configuration. Note that the boundaries shift as the amount of variation in the neural connections increases. Hence, excitation combinations near these boundaries are unstable.

Details of how the controllers respond are as follows:

- *Biological controller:* As the amount of variation in neural connections increases, more failing runs appear on the left of the boundary. However, more

[9]The largest variability value obtained is 9.51 at excitation combination (0.5, 0%) with ±20% variation in neural connections. Under these parameters, the efficient controller switches from an original speed of -0.0065 m/s to positive speeds such as 0.053 m/s.

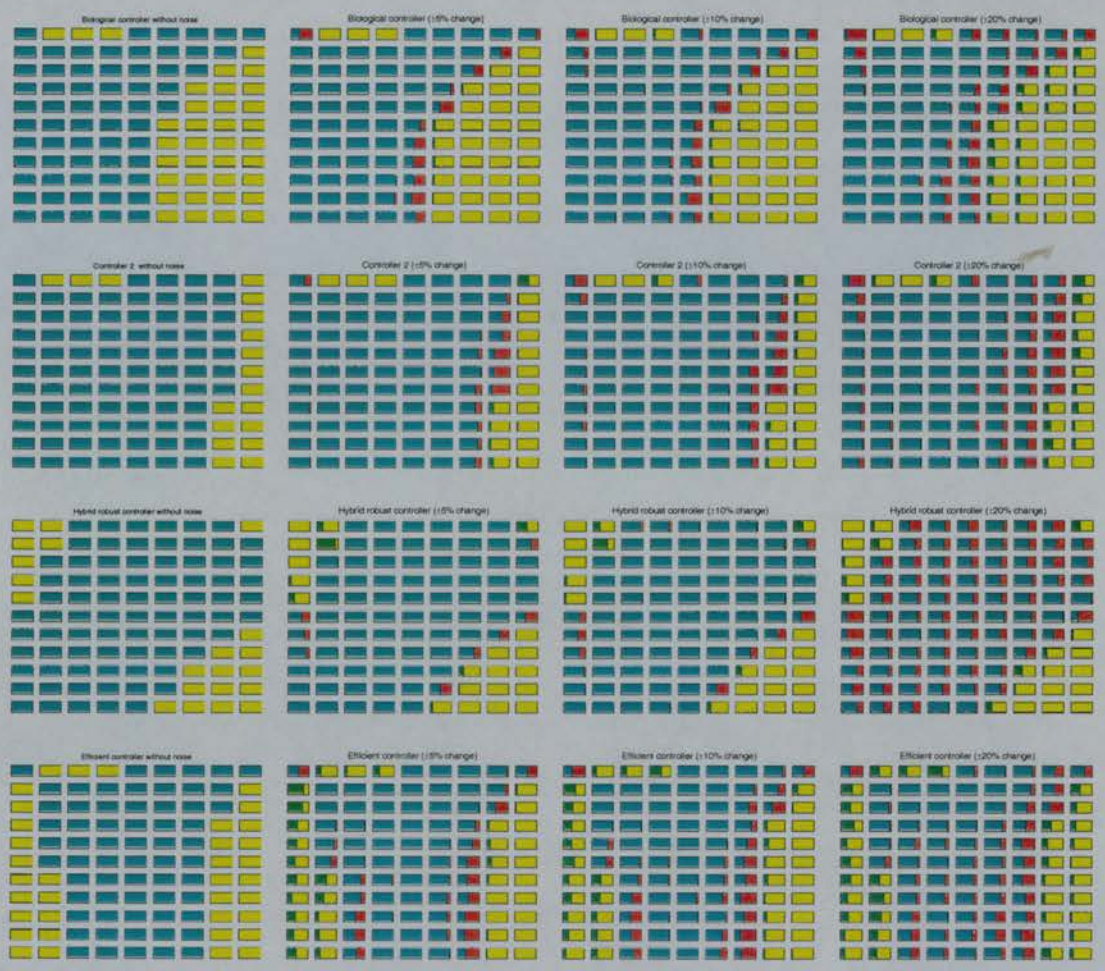


Figure 7.5: Analysis 1: Ability to produce forward swimming. Response of each controller under different levels of noise in both segmental and intersegmental connections. For the matrices in the first column, cyan boxes represent the excitation combination which produces a swimming speed above zero. Boxes in yellow represent invalid speeds (i.e. speeds less than or equal to 0). For the rest of the matrices, green represents success and red represents failure. In this analysis, success means that the swimming speed is above zero. Cyan represents neutral. The area which the green color occupies in any excitation grid is proportional to the number of successes among the 25 runs. Similarly, the area the red color occupies is proportional to the number of failing runs. In each matrix, the first column represents excitation of 0.2 while the last column represents excitation of 1.0. Similarly, the first row represents no extra excitation while the last row represents 200% extra excitation.

successful runs appear on the right. Without extra excitation, the amount of failing runs increases with variation. (Refer to the first row of the four matrices in the top line of Figure 7.5.)

- *Controller 2:* Behaviour is similar to that observed in the biological controller, with the addition that the number of failures increases with noise at the lower end of global excitation 0.2. (Refer to the first column of the four matrices in the second line of Figure 7.5.)

- *Hybrid robust controller:* This controller is the least able to maintain forward swimming when there is variation in the neural connections. With at most $\pm 20\%$ change in neural connections, the number of failures increases at almost all the excitation combinations investigated. Hence, compared with the other three controllers, it is relatively unstable. Note that at the lowest ends of the global and extra excitation ranges (top left corner of the four matrices in the third line of Figure 7.5), more successful runs are achieved with the presence of noise. This implies that the presence of noise allows the model lamprey to swim forward even with very low global and extra excitations.

- *Efficient controller:* This controller behaves similarly to the biological controller except that there are three boundary regions. Increasing the amount of variation in connections led to more successful runs at both ends of the global excitation range (i.e. 0.2, 0.3 and 0.9, 1.0). It is interesting to note that the number of failing runs at the boundaries is fairly constant despite an increase in the amount of noise. The third boundary appears at no extra excitation, where the number of successful runs increases across the global excitation ranges from 0.3 to 0.5. (Refer to the top left corner of the four matrices in the last line of Figure 7.5.) Note that the number of failures in the middle of the excitation grids increases with the level of noise.

Analysis 2: Optimization of the neural configuration

The results from the second analysis are shown in Figure 7.6. (For a zoom-in of the figures, refer to Appendix F.)

Figure 7.6: Analysis 2: Optimization of the neural configuration. Response of each controller under different levels of noise in both segmental and intersegmental connections. For the matrices in the first column, cyan boxes represent the excitation combination which produces a swimming speed above zero. Boxes in yellow represent invalid speeds (i.e. speeds less than or equal to 0). For the rest of the matrices, green represents success and red represents failure. In this analysis, success means that the new speed is higher than the original speed. Failure means that the new speed is lower than the original speed. Cyan represents neutral. The area which the green color occupies in any excitation grid is proportional to the number of successes among the 25 runs. Similarly, the area the red color occupies is proportional to the number of failing runs. In each matrix, the first column represents excitation of 0.2 while the last column represents excitation of 1.0. Similarly, the first row represents no extra excitation while the last row represents 200% extra excitation.

The figure indicates that generally speaking, the evolved controllers are more optimized for producing high swimming speeds at different excitation combinations than the biological controller. This can be explained by the fact that the evolved controllers were created with fitness measures which reward the ability to produce high swimming speeds. For the biological controller, there is a region (extra excitations below 100%) in which its neural configuration is not optimized for high speed (lots of green boxes across the global excitation range). Similarly, for both controller 2 and the efficient controller, the original speeds at extra excitations of 20% and below are generally lower than those achieved after variation in neural connections (lots of green in the first and second rows of the matrices located in the second and fourth lines of Figure 7.6). This implies that the neural configurations of these two controllers are not optimized for phase lags that are very small. This is acceptable as extremely small phase lag corresponds to almost no travel wave and hence low swimming speed. Thus, cases like this are not important when comparing the controllers in terms of the maximum speed they can achieve. As with Analysis 1, some of the excitation combinations which did not produce forward swimming with the original neural configurations end up producing forward swimming after the neural connections are varied.

Details of how the controllers respond are as follows:

- *Biological controller:* As the amount of variation in neural connections increases, fewer runs are able to achieve higher swimming speeds than the originals (the amount of green decreases as the variation in neural connections increases). Meanwhile, some of the excitations which did not produce forward swimming with the original neural configuration end up swimming forward. (Compare the excitation combination grids which are colored yellow in the first matrix with the corresponding grids in the second to fourth matrices in the top line of Figure 7.6.) Note that although some of these combinations lead to faster backward swimming when there is no extra excitation, most of the runs produce higher forward swimming speeds than the originals. This corresponds to the fact that the intersegmental couplings of the original neural configuration favour forward swimming. Generally speaking, the biological controller tends to be more optimized at lower global excitations with large extra excitations. Given that the

frequency of oscillation and phase lag are proportional to the global and extra excitations respectively, the performance of the biological controller is optimal when it is oscillating at a low frequency and when the phase lag is large.

- *Controller 2:* At extra excitations of 20% and below, this controller is less optimized across the range of global excitations under investigation (lots of green in the first and second rows of the matrices in the second line of Figure 7.6). Note that at large global excitation (0.8 and 0.9) and extra excitation (over 20%), almost all the runs produce lower swimming speeds than the original (i.e. the seventh and eighth columns are almost completely red in the matrix in the second line of Figure 7.6). Generally speaking, controller 2 is optimized for a large range of global excitations (especially at the higher end) when there is enough intersegmental phase lag to propagate the neural wave.

- *Hybrid robust controller:* Unlike the previous two controllers, there is no distinguishable region of excitations (colored green) at which most of the new speeds are higher than the original. Hence, the hybrid robust controller is optimized across a wider range of global and extra excitations than the other controllers. Like controller 2, at large global excitation of 0.9, almost all the runs have lower swimming speeds than the original (the eighth column is almost completely red in the second to fourth matrices in the third line of Figure 7.6). Note that at excitation combinations (0.3, 0%) and (0.3, 20%), the original neural configuration did not produce forward swimming. At 5% change in neural connections, most of the runs at these excitation combinations produce forward swimming. However, as the amount of variation continues to increase, the number of these runs decreases. This implies that these two excitation combinations are close to the threshold for which the original controller would produce forward swimming. However, as the amount of variation continues to increase, the modified neural configurations seem to require more global excitations to propel the lamprey forward. Thus, the number of runs with speed greater than zero decreases. At excitation combination (1.0, 0%), increasing the amount of variation allows the modified controller to accept more excitations before getting saturated. Thus, the area coloured green becomes larger as the amount of variation increases. Al-

though the hybrid robust controller is optimized across a wide range of global and extra excitations, it can be seen that its neural configuration is less optimized than that of controller 2 from the fact that the amount of red in each box of the excitation grid is generally less than in the corresponding boxes for controller 2.

- *Efficient controller:* At the boundary regions (i.e. global excitations 0.2, 0.3, 0.9 and 1.0), increasing the amount of variation increases the number of runs which produce forward swimming (more green occupies the excitation combination grids which could not originally produce forward swimming). Meanwhile, at lower global excitations such as those below 0.3, there are cases in which the controller produces faster backward swimming than the original. The fact that there are more cases of forward swimming than backward swimming implies that the original neural configuration (intersegmental couplings in particular) favours forward swimming. This controller is similar to controller 2 in that it is less optimized at low extra excitation of 20% or below (lots of green in the first and second rows in the second to fourth matrices in the fourth line of Figure 7.6). However, for most of the rest of the excitation grids, the original speeds are higher than the new speeds (lots of red overall). In other words, the efficient controller is optimized for high speed at most of the excitation combinations (except those which correspond to small phase lags - those with extra excitation 20% or below).

Analysis 3: Robustness in swimming speed

Figure 7.7 shows the robustness of the controllers at different excitation combinations and amount of variation. (For a zoom-in of the figures, refer to Appendix F.)

As the area which the red color occupies in each excitation grid is proportional to variability (which is the opposite of being robust), the less red the better. In the discussion that follows, let us consider only the excitation grids which produce forward swimming in the original neural configuration, since excitations in the original configuration which led to negative speeds (or speeds very close to zero) are highly likely to have large variability values because changing the neural connections can lead to forward swimming. Thus, variability at these excitation combinations tends to be very

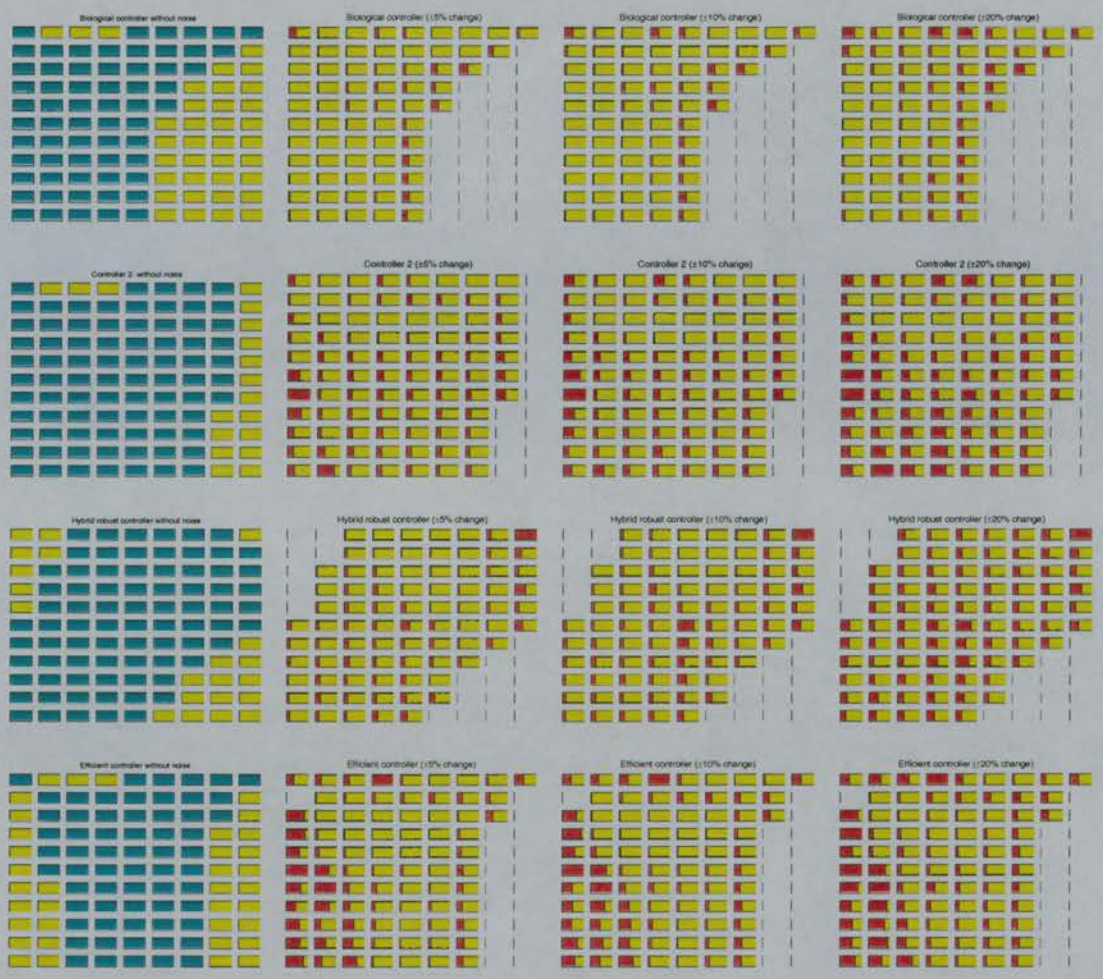


Figure 7.7: Analysis 3: Robustness in swimming speed. Response of each controller under different levels of noise in both segmental and intersegmental connections. For the matrices in the first column, cyan boxes represent the excitation combination which produces a swimming speed above zero. Boxes in yellow represent invalid speeds (i.e. speeds less than or equal to 0). For the rest of the matrices, the area which the red color occupies in any excitation grid is proportional to the average of how much the speeds of the 25 runs change relative to the original speed. The smaller the area, the more robust the controller. To differentiate excitation grids at which the controller is robust, the length of each grid box is standardized to 2. Note that if the original speed is 0, robustness is undefined. This corresponds to those excitation combinations which do not contain any color. In each matrix, the first column represents excitation of 0.2 while the last column represents excitation of 1.0. Similarly, the first row represents no extra excitation while the last row represents 200% extra excitation.

large and can therefore give a false impression of the behavior of the controllers during the comparison. Compared with the evolved controllers, the biological controller is the most robust because there is less red overall. This is followed by the efficient controller. Since the biological controller is hand-crafted from real physiological data, it is reasonable to accept that it is more robust against noise in neural connections than the evolved controllers. Generally speaking, as the amount of variation in neural connections increases, the controllers become less robust (more red). Note that both the biological and efficient controllers become less robust mainly at the excitation combinations in the boundary regions. However, the hybrid robust controller and controller 2 become less robust at most of the excitation combinations under investigation.

Details of how the controllers respond are as follows:

- *Biological controller:* Compared with the evolved controllers, the biological controller has the largest continuous block of excitation combinations which correspond to good robustness (the block is amost completely filled with yellow at the excitation combinations which produce forward swimming in the original neural configuration. Refer to the second and third excitation grid matrices in the first line of Figure 7.7). This block is roughly defined for global excitations from 0.2 to 0.5 and extra excitations from 20% to 200%. Although increasing the amount of variation in neural connections reduces the size of this block, the controller is still relatively robust when compared with the evolved controllers (there is less red across the excitation combinations under investigation - especially for those with global excitation below 0.5). Thus, the biologically plausible controller is very capable of sustaining random variation in neural connections, which is encouraging as this is an important property required for the animal if it is to survive in the real world.

- *Controller 2:* The size of the continuous block with good robustness is smaller than that of the biological controller. The block is defined roughly from global excitations 0.3 to 0.8 and extra excitations from 40% to 50%.

- *Hybrid robust controller:* Again, there is a continuous block of good robustness, defined from global excitations 0.6 to 0.8 and extra excitations below 60%. This

block disappears all together when the amount of variation is increased to $\pm 20\%$, but as this controller was evolved for robustness against variation in body scales rather than neural configurations, one should not expect it to be any more robust than the others.

- *Efficient controller:* The block of good robustness is roughly defined for global excitation ranges from 0.3 to 0.7 and extra excitation ranges from 40% to 80%. Even when the amount of variation is increased to $\pm 20\%$, this controller is still relatively robust. (For example, compare the excitation grids at global excitation 0.7 and extra excitations from 40% to 80% with those of other controllers. The amount of red is less for the efficient controller than for the other controllers.)

7.3 Discussion of methods

In Experiments 1 and 2, I used the excitation combination which corresponds to the maximum swimming speed as the brainstem inputs in each CPG under investigation. Then I observed how random variations in neural connections can affect the original swimming speed. Since the maximum speed can be used as an indication of the best performance the fish can accomplish, it is reasonable to use both this and the corresponding inputs from the brainstem as the basis of comparison. Given that it is more difficult to maintain swimming speed when both the segmental and intersegmental connections are subjected to random variations, I repeated Experiment 2 across different excitation combinations as a worst case analysis of how the controllers behave with different inputs from the brainstem (Section 7.2). As there is no fixed definition of robustness, I conducted three different types of analyses on the same set of experimental data. Given that the analyses are complementary to each other, a comparison of the results from these analyses allows me to gain a better understanding of the behavior of the controllers. In the first analysis, I discovered that for each controller, there are excitation combinations at which the controller becomes unstable. Since these excitations tend to be next to each other, a continuous group of such excitations is called the "boundary region". In the boundary region, the controller is very sensitive to variation in neural connections. Even small changes can easily cause the lamprey to switch

from forward to backward swimming (or vice versa). Note that one may consider the definition of "working" as a swimming speed above zero to be inappropriate because an extrememly small speed (such as 0.00001 m/s) has no significant meaning. Nevertheless, Analysis 1 provides an explanation of the unusual behaviour of the controllers at these boundary regions in subsequent analysis. (For example, explanation of large variability at boundary region in Analysis 3.)

In the second analysis, I used magnitude as the basis of comparison rather than simply whether the fish is swimming forwards or backwards (before and after random variation in neural connections). At each excitation combination, if the variation in neural connections generally leads to a reduction in speed, the original neural configuration is optimized at that specific combination. However, if the speed increases or decreases about half of the time, the neural configuration at that excitation combination is not optimized. Experimental results clearly show that the biological controller is not optimized for fast swimming to the same extent as the evolved controllers.

The reason why the same thing was not done for Experiment 4 is because swimming efficiency is determined by more factors than these - such as swimming speed and mechanical wavelength - and it is therefore more difficult to relate the efficiency change to the presence of noise.

In Section 7.1, I compared the behavior of the controllers across the four experiments. Since Experiments 3 and 4 deal with maximum efficiency (which can be defined only if the speed is positive), negative speeds were not considered when I computed the average speeds at each noise range in Experiments 1 and 2. Similarly, swimming speeds equal to zero were not counted as they correspond to speeds which cannot be measured. If all the speeds were positive, I used the speed achieved in all 100 runs of the neural and mechanical simulations (with random seeds in the random number generator) to calculate the average. Note that there could still be measurement errors due to out-liers. In other words, there is a possibility that a large amount of variation in neural connections could mean that the averaged values become correspondingly less accurate. Even though the standard deviations are provided to allow a comparison of the data, the problem caused by out-liers should still be borne in mind.

When I computed the average amplitude, frequency and phase lag for Figures 7.3

and 7.4 (in Subsection 7.1.4), runs which corresponded to invalid speeds or efficiencies are removed. (Refer to the histograms in Appendix G for the number of runs used to calculate the averages.) Note that there is a tendency for the number of runs used to calculate the averaged phase lag to be less than the number of runs used to calculate other averages. This implies that phase lag is very sensitive to noise in neural connections.

In order to avoid the problem caused by ignoring the negative speeds, I calculated the variability (which includes the negative speeds as part of the calculation) of the controllers in Analysis 3 as a possibly more accurate measure of robustness.

As for the amount of stochastic variation, a noise range from 10% to 50% is reasonable since it corresponds to a maximum of $\pm 5\%$ to $\pm 25\%$ change in the segmental connection weights and/or intersegmental couplings. Given that an integer is used to represent an intersegmental coupling, each extension can be varied from 0 to ± 3 segments. Note that as the calibrated connection strengths depend on the number of extensions to neighboring segments, the calibrated segmental connection strengths for the controllers in Experiments 2 and 4 may vary more than their counterparts in Experiments 1 and 3.

At the beginning of each simulation, I randomly varied the neural connections. All segments were subjected to the same amount of noise. In other words, the extensions from each source neuron to the corresponding target neuron in the rostral/caudal neighbours were the same regardless of its location. Although this appears to be less realistic since different parts of the real CPG can be subjected to different amount of variations, the resulting calibrated connection strengths are actually different along the CPG because the end segments receive fewer connections than those in the middle of the network. Given that evaluating the controllers' performance under such systematic random variation in neural connections could be considered more of an exploration of the local optimality in the search space (rather than robustness), my approach is still useful. Future work will involve giving each segment independent noise so that even segments in the middle of the CPG can be varied differently from each other. This will be more realistic and sensible from the biological point of view.

Left-right symmetry in neural connections is imposed just as in the other experi-

ments described in this dissertation. Otherwise, there could be a lot of disturbances in swimming performance. For example, turning is more likely to occur as the connection weights on one side of the CPG can be stronger than those on the opposite side, and in the worst case, the CPG might not even oscillate.

Finally, due to the nature of random variations, a connection may vary more when it is subjected to $\pm 5\%$ rather than $\pm 10\%$ of variation. Similar explanations can be made for the investigation of robustness in swimming efficiency against noise range in Experiments 3 and 4.

7.4 General discussion of the four controllers

In this section, I discuss the behavior of the controllers in terms of the optimization and robustness of their neural configurations. Through comparative analysis, some of the features of intersegmental couplings which may be responsible for these two properties are discussed.

7.4.1 Optimization and robustness of the controllers

The results from Analysis 1 show that both the biological controller and controller 2 are robust across the ranges of global and extra excitation combinations under investigation (see Figure 7.5). The hybrid robust controller is, on the other hand, the least robust. With $\pm 20\%$ variation in both segmental and intersegmental connections, the number of failing runs increases significantly at almost all the excitation combinations under study. The performance of the efficient controller is somewhere between these two groups of controllers. With the increase in variation of any neural connection, there is an increase in the number of both successful and failing runs.

Figure 7.6 shows that in general, the biological controller is optimized for the production of high swimming speeds at low global excitations and high extra excitations. The evolved controllers, on the other hand, are optimized for a larger range (especially at the higher end) of global excitations when there are sufficient extra excitations. At the excitation combinations where the maximum speed of these controllers occurs, the

biological controller is the least optimized (compare the pairs of numbers in the first row with the others in Table 7.7). The evolved controllers, on the other hand, are all optimized. Thus, the GA did a good job of optimizing the performance of the controllers.

Controller	5%	10%	20%	Excitation combination
Biological	[7, 18]	[8, 17]	[3, 22]	(0.6, 200%)
Controller 2	[1, 24]	[2, 23]	[1, 24]	(0.9, 40%)
Hybrid robust controller	[1, 24]	[1, 24]	[0, 25]	(0.9, 0%)
Efficient controller	[1, 24]	[1, 24]	[1, 24]	(0.9, 40%)

Table 7.7: Local optimum of each controller at different amounts of variation. The first number in the brackets represents the number of runs with higher speed than the original. The second number in the brackets represents the number of runs with lower speed than the original. The larger the second number, the closer the controller is towards the local optimum. Note that the measurements are taken at excitation combinations which correspond to the maximum swimming speeds achieved by the controllers.

One important point worth mentioning is that as we have no idea about which neural configuration is the most optimized (globally) to achieve high swimming speed, controllers which I consider to be optimized in this thesis are actually optimized locally in the sense that variation in neural connections decreases the swimming speeds. Recall that in this dissertation and Ijspeert [1998], the controllers were evolved to optimize swimming performance. Also, the hybrid robust controller evolved in Or et al. [2002] was optimized for robustness against variation in body parameters. The results here indicate that these controllers turned out to be suboptimal in terms of robustness against variation in neural connections (which I did not try to optimize).

Figure 7.7 shows that the biological controller and the efficient controller are relatively robust across the global and extra excitation combinations under investigation. Controller 2 and the hybrid robust controller are less robust. Generally speaking, at the excitation combinations where the maximum speed of these controllers occurs, the biological controller is the most robust (the variability across different amounts of variation tends to be equal to or less than the corresponding one for the other controllers

listed in Table 7.8.). It is followed by controller 2, the hybrid robust controller and then the efficient controller.

Controller	5%	10%	20%	Change in variability
Biological	0.47	0.43	0.39	-0.08
Controller 2	0.47	0.43	0.62	0.15
Hybrid robust controller	0.41	0.47	0.76	0.35
Efficient controller	0.65	0.64	0.70	0.05

Table 7.8: Robustness (in terms of variability) of each controller at different amounts of variation. Note that the measurements are taken at excitation combinations at which the controller produces its maximum swimming speed. Change in variability is calculated as the difference in variability between 5% and 20% of variations.

As the amount of variation increases, the variability of the biological controller decreases. However, the variability of each evolved controller increases. This means that the biological controller becomes more robust than the other controllers when the amount of noise in the system is large. Given that the efficient controller and the biological controller have the least change in variability from 5% to 20%, their optimum could be at the top of a flat mountain, whereas for controller 2 and the hybrid robust controller, their optimum could be at the top of a peak. In other words, slight variation in neural connections can lead to a greater speed change for both controller 2 and the hybrid robust controller.

7.4.2 Features of intersegmental couplings related to the optimization, robustness and efficiency of the swimming controllers

As mentioned throughout the thesis, although there have been some experimental data on the intersegmental couplings of the lamprey central pattern generator, details of how the segmental oscillators are connected or what are the underlying mechanisms responsible for intersegmental coordinations are still imperfectly understood [Buchanan,

1992, Hellgren et al., 1992, Wadden et al., 1997, Ullström et al., 1998, Buchanan, 1999, Hellgren et al., 1999a,b, Miller and Sigvardt, 2000].

Table 7.9 presents, to the best of my knowledge, the latest findings on synaptic connectivity known in the real lamprey [Ullström et al., 1998].

Type of synapse			One local CPG	Distributed CPGs		
Presynaptic unit	Postsynaptic unit	type	w_0	λ_r	λ_c	w
EIN	EIN	ex	10.0	2	2	2.0
EIN	CIN	ex	10.0	2	2	2.0
CIN	EIN	inh	2.5	2	20	0.11
CIN	CIN	inh	2.5	2	20	0.11
BS	EIN	ex	2.0	-	-	2.0
BS	CIN	ex	2.0	-	-	2.0

Table 7.9: Latest findings on intersegmental couplings. λ_r and λ_c respectively represents the extensions from each segment to its rostral and caudal neighbours. w_0 denotes the connection weights between the presynaptic and postsynaptic neurons within a segmental oscillator. When the segmental oscillators are coupled, the connection weights can be calculated using $w = w_0/(1+\lambda_r+\lambda_c)$. (Reproduced from Ullström et al. [1998] with permission.)

Unlike the Ekeberg neural model used in this thesis, the lateral inhibitory interneuron (LIN) found in the lamprey is not included because it is not primary important for burst termination [Hellgren et al., 1992, Ullström et al., 1998]. According to Hellgren (personal communication):

> Regarding the LIN it seems they are more involved in turning and steering. LINs also exist only in the most rostral part of the cord. But there might be functionally similar ones (but smaller) locally with much shorter projections, thus a certain role of ipsilateral inhibition in burst termination can not be excluded (but for the moment we don't use them). In all earlier models that use them LINs are needed for ipsilateral burst termination.

In regard to the CIN neurons, the current findings is that they project up to 20 segments caudally. This is twice as long as is proposed in the Ekeberg model. It is

known that the CIN neurons have thinner projections in the rostral direction. However, according to Hellgren (personal communication):

> there exist other types of commissural interneurons with shorter caudal as well as rostral projections as well and these might be equally important for coordination... We thought that the strength should decrease with caudal distance so that the inhibition is strongest locally within a few segments.

In this dissertation, I have studied some of the properties essential to the survival of the lamprey. In particular, I investigated firstly robustness in maximum swimming speed against variation in body parameters (Chapters 4 and 5), and secondly swimming efficiency (Chapter 6). Thirdly, in this chapter I analysed the robustness in swimming speed and efficiency against noise in neural connections . As part of this investigation, optimization of the swimming controllers was studied.

A comparative analysis of the neural configuration for each controller should provide a better understanding of the features in the neural networks that are related to these properties. As these properties are important to the real lamprey, my results will potentially provide a better understanding of intersegmental couplings and the general principles underlying lamprey swimming. For instance, a comparison of the neural organization of the efficient (*bigmin run9*) controller evolved in Chapter 6 and that of the real lamprey shows that there tends to be short (2 to 3) symmetrical connections from the excitatory EINl neurons to the EINl neurons in both rostral and caudal directions. Moreover, the connections from the inhibitory CINl neurons to the CINr neurons tend to favour the caudal directions. Interesting, these tendencies are present in the Ekeberg model too. (Compare the connection weight matrix shown in Figure C.3 with Tables 7.9 and 2.1.) Thus, they are likely to be important to the lamprey. However, at present, the kind of behavior(s) which relates to these tendencies in intersegmental couplings is unknown. Given that the swimming efficiency of both the real and model (driven by the efficient controller) lampreys are the same, it may be that these features are required for the lamprey to swim efficiently. Through collaboration with neurobiologists, my hypothesis can be confirmed in real biological experiments. To summarize, I use GA as an alternative approach to understanding biology via synthesis.

The four controllers investigated in this chapter can be categorized into two groups. The first group involves the Ekeberg biological controller, controller 2 and the efficient controller. The segmental oscillators of these three controllers are the same (i.e. controller 2 and the efficient controller are composed of 100 copies of the same segmental oscillator which makes up the Ekeberg biological controller). The hybrid robust controller, on the other hand, is different: both its intra- and inter-segmental couplings were evolved by GAs in [Or et al., 2002]. Given that the first group of controllers share the same segmental oscillator, any difference in behavior is most likely to have been caused by the differences in intersegmental couplings. Note that although this assumption simplifies the comparisons, the same neuron (at any specific segment) can receive different inputs within different controllers due to the ways the neurons are connected (refer to Subsection 3.1.1 for further details).

In the discussion that follows, I assume that the neural organization of the biological controller (hand-crafted by Ekeberg) shares some of the essential features of the real biological controller. Thus, findings gained through comparison of this controller with its evolved counterparts can be transferred to the biological field for a better understanding of the real lamprey's locomotion neural networks. I also assume that the properties which I mentioned previously are reasonable and that the real biological neural networks are evolved through natural evolutions to favour these properties. Note that as there are many different potential neural configurations which allow the artificial lamprey to behave similarily (as mentioned in Ijspeert [1998]), confirmation from real biological experiments is crucial to the hypothesis made in this chapter.

In order to allow a better comparison of the neural organizations of different controllers, I express the number of rostral and caudal extensions between each set of connected neurons in terms of a stacked bar graph as shown in Figure 7.8. At each pair of neural connections, the stacked bar graphs of the four controllers are aligned side-by-side. Based on the results from Section 7.2, each set of stacked bar graphs is arranged in the following order: biological controller, efficient controller, controller 2 and the hybrid robust controller.

In order to simplify the wording in the discussion that follows, I use $extent(rostral(X))$ and $extent(caudal(X))$ to represent the length of the intersegmental connections of

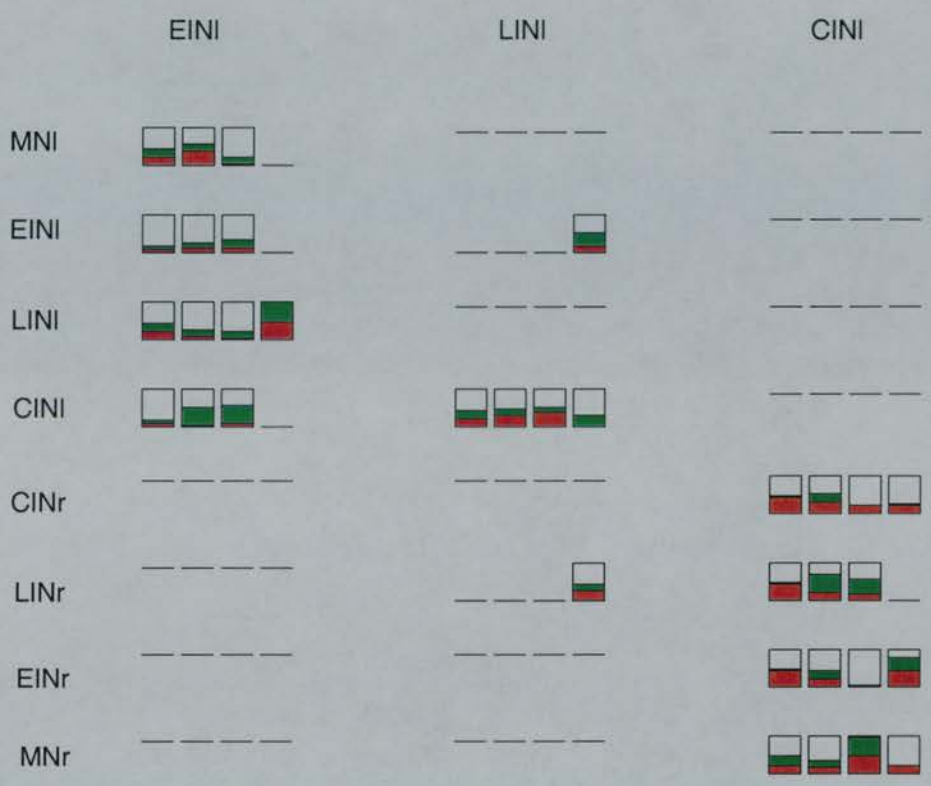


Figure 7.8: Intersegmental couplings of the biological controller, efficient controller, controller 2 and the hybrid robust controller. At each stacked bar graph, the area in green is proportional to the length of intersegmental connections in the rostral direction while the area in red is proportional to the length of intersegmental connections in the caudal direction. Note that the connections from the LIN and CIN neurons are inhibitory.

controllers X in the rostral and caudal directions respectively. Note that X can be a list of any of the four controllers (simplified as bio, con2, robust, eff) mentioned above.

Biological controller vs. efficient controller and controller 2

From the results of Analysis 2 in Subsection 7.2.3, I concluded that the biological controller is not as optimized as the evolved controllers. A comparison of the neural configuration of this controller with that of the efficient controller and controller 2 shows the following differences:

- EINl to LINl: $extent(caudal(bio)) > extent(caudal(eff, con2))$.
- EINl to CINl: $extent(rostral(bio)) << extent(rostral(eff, con2))$.
- CINl to LINr: $extent(rostral(bio) << extent(rostral(eff, con2))$.

  However, $extent(caudal(bio)) >> extent(caudal(eff, con2))$.
- CINl to EINr: $extent(caudal(bio)) > extent(caudal(eff, con2))$.

Biological controller and efficient controller vs. controller 2

From the results of Analysis 3 (in Subsection 7.2.3), I found that both the biological controller and efficient controller are more robust across a wide range of global and extra excitation combinations under investigation. The performance of controller 2 is fair while that of the hybrid robust controller is the worst. A comparison of the neural organizations of the first two controllers with controller 2 may reveal intersegmental connections which allow better robustness (measured in terms of variability) in swimming speed against noise in both segmental and intersegmental connections. The main differences are shown as follows:

- EINl to MNl: $extent(caudal(con2)) << extent(caudal(bio, eff))$

  Note that the number of extensions in the rostral direction is similar in the three controllers.
- EINl to LINl: $extent(caudal(con2)) << extent(caudal(bio, eff))$.
- CINl to CINr: $extent(rostral(con2)) < extent(rostral(bio, eff))$.
- CINl to EINr: $extent(caudal(con2)) << extent(caudal(bio, eff))$.
- CINl to MNr: $extent(rostral(con2) > extent(rostral(bio, eff))$.

  Also, $extent(caudal(con2)) >> extent(caudal(bio, eff))$.

Efficient controller vs. other controllers

From Table 6.3, the efficient controller has the highest efficiency. It is followed by the hybrid robust controller and then controller 2 and the biological controller. A comparison of the neural configuration of this controller with that of the other controllers shows the following differences and similarities:

- CINl to CINr: $extent(rostral(eff)) >> extend(rostral(others))$.
- CINl to EINr: $extent(rostral(eff, robust)) >> extent(rostral(bio, con2))$.

These relationships clearly show that certain features of the intersegmental couplings are related to the optimization, robustness and efficiency of the swimming controllers. In order to test my hypothesis, more in-depth studies are required. For example, as the first step, one might increase the length of rostral extensions from the EINl to CINl neurons within the biological controller to match that of controller 2 or the efficient controller. If the biological controller becomes more optimized across a wider range of excitations (as exhibited by the matched controller), it could then be inferred that longer rostral extensions from the EINl to CINl neurons correlates with better optimization at different excitation combinations. Similarly, one could increase the caudal extensions from the CINl to EINr neurons of controller 2 to match that of the biological controller. If the modified controller 2 becomes as robust as the biological controller (across a similar range of global and extra excitation combinations), then it could be inferred that a long caudal extension from the CINl to EINr neurons is required for a controller to be robust across a wide range of excitations.

As for high swimming efficiency, it seems that rostral connections from the CINl neurons to the CINr and/or the EINr neurons may play a role. For instance, although the hybrid robust controller is made up of a different kind of segmental oscillator, its CINl neurons have connections to the EINr neurons of 8 rostral neigbours. Similarly, the corresponding length in the efficient controller is 5. Unlike these two highly efficient controllers, the biological controller and controller 2 (whose efficiencies are close to each other) have only one rostral extension. Thus, long rostral extensions from the CINl neurons to the EINr neurons may correlate with high efficiency. One way to test

this hypothesis would be to vary the connection length from the CINl neurons to the EINr (of the efficient controller) and then observe if the swimming efficiency scales with connection length. Another observation is that the efficient controller has much longer extensions from the CINl neurons to the CINr neurons in the rostral direction while the other controllers have less than one in the corresponding connection. Thus, varying the length of rostral connections from the CINl neurons to the CINr neurons may change the swimming efficiency as well.

Note that optimization, robustness and efficiency may be emergent properties of the complex interactions among the neurons. Thus, varying the intersegmental couplings one at a time may not lead to the desired behavioral change. For example, although the hybrid robust controller has longer rostral couplings than the efficient controller, its efficiency is less. This may be due to its intra-segmental connections and/or longer couplings between the CINl and EINr neurons in the caudal direction. Thus, lots of experiments may be required before a relationship between intersegmental couplings and swimming performance is found. Hopefully, through systematic and intelligent application of my approach to many different variations of each controller, some kind of pattern relating to these properties will emerge. If simulation results turn out to be satisfactory, real biological experiments which parallel some of the above mentioned experiments should then be carried out for confirmation. This can be accomplished through a combination of anatomical, physiological and lesion experiments.

7.5 Summary

This chapter investigated the robustness of the four most interesting central pattern generators encountered in previous chapters. Since there is no universal definition of robustness, I used several different criteria to evaluate the controllers. Two sets of experiments were conducted.

In the first set of experiments, I investigated how varying the segmental connection weights and/or intersegmental couplings affects the maximum swimming speed and efficiency. I observed that controller 2 is the most robust across different experiments. It is followed by either the biological controller or the efficient controller (depending on experiments). The performance of the hybrid robust controller evolved in Or et al. [2002] is the worst. Most importantly, although the absolute swimming performance (in terms of maximum speed and efficiency) of the biological controller is not as good as the artificially evolved controllers (it is actually quite good already), it is relatively robust against noise in the system. The robustness of the model CPGs might be explained by the robustness of the real biological circuit due to natural evolution.[10]

In the second set of experiments, I investigated what effect varying both the segmental and intersegmental connections has on the swimming speed across the different excitation combinations under investigation. Three different measurement methods were used. In the first analysis, I found that the controllers differ in their ability to maintain forward swimming. For each controller, there are boundary regions (sets of excitation combinations) at which the controller becomes unstable. Compared with the others, the hybrid robust controller is the least stable. In the second analysis, I found that the evolved controllers are more optimized for producing high swimming speeds at different excitation combinations than the biological controller. In other words, the GAs successfully generated controllers which produce high swimming speed overall. Generally speaking, the evolved controllers tend to be more optimized across a larger range of global excitations (especially at the higher end) while the biological controller is optimized at lower global excitations and higher extra excitations. The final analysis compares the robustness of the controllers in terms of variability. The biological controller and the efficient controller are equally robust across a wide range of excitation

[10]Recall that the biological controller is hand-crafted by Ekeberg based on physiological data.

combinations. The former is better at global excitation ranges from 0.2 to 0.5 while the latter is better at global excitations from 0.3 to 0.7. The performance of controller 2 is fair while that of the hybrid robust controller is the worst.

Finally, through comparative analysis, I have provided a list of intersegmental couplings which may potentially be responsible for the optimization, robustness and efficiency of the controllers. Interaction with neurobiologists will be necessary to confirm my hypothesis and findings.

Chapter 8

Conclusions

So far, we still have an imperfect understanding of the intersegmental couplings which allow the lamprey to achieve efficient swimming and adaptation to change in its body parameters. The main goal of this thesis is to investigate the use of the Genetic Algorithm as a tool for creating robust and efficient swimming controllers for a simulated model lamprey (which was made to approximate the size and shape of the real one). As part of the investigation, the performance of the evolved controllers was compared with that of the biological controller hand-crafted by Ekeberg. Results indicate that the best evolved controller, the efficient controller, is able to perform better than the hand-crafted controller (see Table H.1 in the appendix for a brief summary). Also, through a comparison between the population-based GA and other classical optimization techniques, I have shown that the GA, although maybe not the most efficient method overall, is still a more appropriate design tool for automatically creating a swimming controller for the simulated lamprey. It produces a controller which has the ecologically significant properties of being robust and efficient, as well as being able simply to swim. This is the key point of the thesis.

The behavior of the fitness function as a function of generation for the evolution of the efficient controller (shown in Figure 6.9) indicates that the evolved efficient controller created at generation 60 is near, but not at, the local optimum. Increasing the number of generations from 60 to 150 increases the maximum fitness by 0.007 (about 8%). Thus, better controllers might still be evolvable. However, considering that the evolved efficient controller is able to produce a higher speed and efficiency

than any other controller, and since, at the time of the experiments, there was no way of telling when and whether the maximum fitness will ever increase significantly, I stopped the GA at generation 60.

Towards the end of the thesis, a list of the features of intersegmental couplings which may potentially be responsible for the robustness and efficiency of the swimming CPGs was proposed. Since the efficient controller satisfies these properties (which are important for the survival of the lamprey), it is worth considering using the neural connections of this controller as the inspiration for carrying out real biological experiments through collaboration with neurobiologists. The findings obtained throughout this dissertation have the potential to provide a better understanding of intersegmental couplings and the general principles underlying lamprey swimming. Even if the neural organization of this controller turns out to be just one possible configuration that has these important properties, the results from this dissertation can still be applied to control an artificial lamprey (as demonstrated in this dissertation) and possibly a robotic one (in the near future).

8.1 Discussion

This section discusses some of the choices I made throughout the dissertation. In particular, I discuss the reasons for using the GA as a design method rather than other potential alternatives. This is followed by a discussion of the reasons for modeling the lamprey locomotion neural networks at the connectionist level. Finally, the reasons for choosing the Ekeberg mechanical model of lamprey swimming in stationary water rather than the more detailed model proposed by Carling et al. [1998] (fluid dynamics of water surrounding the model lamprey is taken into consideration), are explained.

8.1.1 Reasons for using GA as a design method

In this dissertation, I used genetic algorithms to evolve locomotion controllers for the lamprey. The reasons for using the GA rather than other learning algorithms such as backpropagation or reinforcement learning (RL) are as follows.

In backpropagation (supervised) learning, once the input signals have been propagnated through the network, the corresponding outputs are calculated. The connection weights among the neurons in the network are adjusted to reduce the difference between the network's outputs and the desired output. Although this learning paradigm has been successfully used to solve some of the problems in pattern recognition and time series prediction, its application is limited by the requirement that the error function be continuous and differentiable. Furthermore, the desired outputs have to be known in advance. Since the swimming of the lamprey is an emergent behavior resulting from complex interactions between the body and the water, there is no direct mapping between command signals from the brainstem and movements of each body part. One advantage of the GA is that the designers only need to specify the desired behavior at a higher level. It does not require the fitness function to be continuous and differentiable. Furthermore, compared with backpropagation, it is less likely to be trapped in a local optimum (due to the mutation operator and its parallel nature of exploring different parts of the search space at any time). Thus, GA is more suitable than backpropagation for creating swimming controllers.

According to Barto [1995], reinforcement learning is based on the idea that if an action is followed by a satisfactory state or an improvement of the current state, then the tendency to produce this action is reinforced. In such a learning system, a critic is present to provide a feedback value based on the control signals and the resulting behavior. This paradigm suffers from the credit assignment problem. For example, given a value provided by the critic as an evaluation to the swimming performance, there is no way to determine which of the components of the lamprey body parts are responsible for the evaluation, since, as mentioned above, anguilliform swimming is a complex emergent behavior. Even if this problem is solved, more than one critic may be required for each body segment if anguilliform swimming is the only desired swimming mode. Although the RL can be applied to problems which require learning the optimal policy (such as the Markov decision process), this kind of algorithm requires a lot of time. Therefore, it is an undesirable method for creating swimming controllers.

Given the problems associated with backpropagation and reinforcement learning, as well as the merits that the GA offers, I chose the GA as a learning method despite

the fact that it requires a lot of computation time.

8.1.2 Reasons for choosing the connectionist level of modeling

As mentioned in Section 2.3, several models of the lamprey swimming CPG have been proposed. These models are divided into three main groups according to level of complexity: detailed biophysical models, connectionist models and mathematical models of chains of coupled oscillators. The reasons I chose the connectionist model rather than models from the other two groups are as follows:

1. Although simulating detailed biophysical models may allow one to better relate the results to biological reality at some levels, the simulations are still computationally expensive, even with the computational power of modern computers. The situation is worse still as the GA requires a large number of neural simulations. Moreover, even with advances in physiological measurement techniques, some neural parameters still cannot be measured. In order to be biologically compatible, some of the neural parameters have to be estimated. This means that the biophysical simulations can become less reliable than reality.

2. As details of cellular properties as well as connectivities of the neurons are not considered in the abstract mathematical models, they are less likely to relate to biological reality. Given that one of the aims of this dissertation is to understand biology via synthesis with evolution algorithms, mathematical models are not chosen.

3. In terms of complexity, connectionist models are simpler than biophysical models but are more complex than mathematical models. This approach to modeling the CPG allows me to focus on the interconnections (such as connection weights) between groups of functionally similar neurons and their effects without worrying too much about the details of all the neuron parameters (such as the membrane and synaptic properties). Results from [Buchanan, 1992, Williams, 1992a, Ekeberg, 1993, Jung et al., 1996, Ijspeert, 1998] show that the connectionist models are capable of producing rhythmic activities similar to those observed

in real physiological experiments. Furthermore, Ekeberg [1993] and Ijspeert [1998] demonstrated that the same connectionist model is capable of controlling the model lamprey to allow it to swim like a real one. Given these merits, and for better comparison of results, I decided to use the connectionist model.

8.1.3 Reason for choosing the Ekeberg mechanical model

Currently there are two 2D mechanical models for the lamprey. One proposed by Ekeberg [1993] (with some parameters later corrected by Ijspeert [1998]) and the other proposed by Carling et al. [1998]. These models were created for different purposes. The Ekeberg model was designed as a means of evaluating the connectionist neural model through the mechanical model lamprey which it controls in the simulated physical world. The performance of the controller can then be evaluated according to the swimming performance of the model lamprey. In order to simplify the computation, water forces are assumed to be inertial. Results indicate that the Ekeberg model is capable of swimming like the real lamprey. As for the Carling model, it was created to investigate self-propelled anguilliform swimming using a computational model which takes the interaction between the model fish's movement and the flow of surrounding water into consideration. As a result, fluid dynamics have to be taken into account. Although this approach can allow a more realistic study of anguilliform swimming, the computation of fluid dynamics increases the simulation time. Furthermore, extra care has to be taken during numerical integration of equations which govern the water and body motions. According to Carling et al. [1998], "a small departure from the correct position of the creature's body, as may well occur during any iterative process, produces a relatively large corrective force arising from the fluid dynamics. Without due care, this process can easily become numerically unstable." Given that I am using the GA to evolve locomotion controllers for the model lamprey, integration error is likely to present a problem, especially during the earlier stage of the evolutions. Since the simpler model has been used by both Ekeberg and Ijspeert to interface between the connectionist neural network and the physical world, I used the same model for better comparison of results. Interestingly, Carling et al. [1998] found that his model lamprey has a swimming efficiency of 0.77, which is very close to the one observed in the real

lamprey and the Ekeberg model lamprey (driven by the efficient swimming controller which I evolved in Chapter 6).

8.2 Contributions

The results from this dissertation provide important contributions to the fields of neurobiology, ALife and robotics.

8.2.1 Contributions to neurobiology

A different approach to the investigation of the intersegmental couplings of lamprey swimming controllers has been presented. Throughout this dissertation, several interesting controllers have been evolved. It is encouraging to find that the most efficient controller allows the model lamprey to swim at the same efficiency as the real lamprey.[1] Also, its neural configuration is optimized for high speed swimming (at least locally in the search space which the GA explored) and robust across a wide range of excitation combinations. Through comparative analysis of the neural organization of these controllers and that of the biological controller, I proposed a list of features in intersegmental couplings which might be responsible for the performance, robustness and efficiency of the swimming CPG (refer to the discussion in Subsection 7.4.2). As these properties are important to the real lamprey, my results will potentially provide a better understanding of intersegmental couplings and the general principles underlying biological motor control, at least for the lamprey. Through collaboration with neurobiologists, the list of features mentioned above can be used as a pointer towards real biological experiments (refer to Subsection 7.4.2 and Section 8.3 for a list of potential experiments and the possible knowledge which we may gain from them). Thus, this thesis has opened up the possibility of new biological experiments to validate my approach of understanding biology through synthesis.

Neurobiologists have been studying lamprey intersegmental couplings for over 15 years. However, many of the details are not known yet. (For a review, see [Miller and

[1] However, the network organization of the evolved CPG may be significantly different from that of the real one.

Sigvardt, 2000].) This is mainly due to the fact that the number of neural connections is enormous, and also partly because the behavior of the real lamprey may change unexpectedly once the neural connections are disconnected. For example, Buchanan [1999] found that "when the spinal cord preparation was cut completely along the midline into two lateral hemicords, the rhythmic activity of fictive swimming was lost." Conducting experiments in computer simulations has the advantage that one can vary either one or a group of neural connection(s) independently to observe how the swimming behavior is affected. Moreover, the experiments can be repeated as often as possible with all the variables under control. Thus, the approach which I proposed in this dissertation should be a good one. It would be fruitful to extend my methodologies to study the virtual salamander (in 3D) reported in [Ijspeert, 2001]. The neural network of the salamander is even less well known but has been hypothesized to be an extension of the lamprey swimming central pattern generator [Delvolvé et al., 1997]. The results may lead to better understanding of amphibian motor control.

8.2.2 Contributions to ALife

Since rhythmic patterns are the building blocks of repetitive behavior [Ijspeert, 1998], the methods of evolving robust and efficient locomotion controllers developed in this dissertation can be applied to create controllers for robots with multiple redundant actuators (see below) or virtual characters in realistic computer animations. For instance, Ijspeert [1998, 2001] created locomotion controllers for a virtual salamander by adding two sets of oscillators to the lamprey swimming CPG. These additional oscillators control the limbs of the salamander during trotting. Similarly, the locomotion controllers for other kinds of animals (such as the aliens in the film *Final Fantasy*, the dinosaurs in *Jurassic Park*, and the monsters in *Lord of the Rings* and *Monsters, Inc.*) can be evolved. As everything can be done using computers which are relatively inexpensive these days, real animatronic models or actors/actresses are not required. This could cut production costs by a great deal. Hence, the results from this dissertation may be beneficial to the movie and video game entertainment industries.

8.2.3 Contributions to robotics

Currently, there are two methods of evolving controllers for robots: online and offline methods [Nolfi and Floreano, 2000]. Each has its own advantages and disadvantages. Although online evolutions can increase the chance that the evolved controllers will be more capable of controlling the robot in the real world, this method requires a huge amount of time. Furthermore, problems such as the possibility of damaging the robot by bad controllers (especially at the earlier stage of evolution) and the need for human intervention (such as placing the robot in its original position at the beginning of each controller evaluation) make offline evolution a better alternative, especially since less time is required to complete a generation. However, the problem with offline evolutions is that controllers which score well in simulations may not perform well in the real world. Jakobi et al. [1995] investigated this issue by evolving obstacle avoidance and light seeking controllers for the Khepera robot under computer simulations with different levels of fidelity. Later, they compared the performance of the controllers in simulations and in the real world. They found that once a good simulation model has been developed, the controllers exhibit similar behavior in both simulation and reality. Furthermore, they have found that controllers evolved in simulation in environment noisier than the real world can generally behave less noisily when downloaded into the robot, and *vice versa*. However, the level of noise in simulation should be just right. It cannot be too different from reality, otherwise controllers which can achieve high scores in simulations can fail completely in reality. (For a review of work on the transition of controllers from simulation to real robot, refer to [Jakobi et al., 1995]. Jakobi [1998] provides a series of more in-depth investigations of the "minimal simulation" approach to evolutionary robotics.)

Given the pros and cons of each method, the best strategy would be to evolve controllers in simulations first and then continue the evolutions online once the best evolved controllers can generate regular oscillating control pulses. Since one of the main problems with simulations is the difficulty in modeling sensors, online evolutions are particular useful when sensory feedback is involved.

The strategy of combining the two methods has been applied successfully in several robotics projects. (Nolfi and Floreano [2000] provides a good introduction to and

overview of evolving controllers for robots.)

The University of Pennsylvania has developed a biomimetic robotic eel (the *REEL*) which can swim autonomously on the water surface [McIsaac and Ostrowski, 1999]. The robot consists of five rigid links with a servo-motor at each joint. Unlike the one described in [Ayers et al., 2000], this robot is untethered and its energy source is on board. One immediate application of my results would be to either evolve swimming controllers using the same fitness function presented in Chapter 6 but with changes in body parameters to match the REEL robot, or simply to carry out online evolution with a future 10 link version of the REEL. As the efficient controller is robust against variation in body parameters and a lot of assumptions made in modeling the REEL robot in simulations are based on [Ekeberg, 1993],[2] it is reasonable to expect that the efficient swimming controller would be able to control the robot. Although the REEL is controlled by a variant of the sinusoidal controller presented earlier in this thesis, a path-planning algorithm is required to steer the robot from place to place [McIsaac and Ostrowski, 1999]. The advantages for using the efficient controller is that based on the experimental results presented in Chapter 6, the efficient controller is much more capable of allowing the model lamprey to swim with high efficiency than is the sinusoidal controller. Furthermore, complex path-planning algorithms are not required because as shown in Ijspeert [1998], the artificial CPG can be made to chase a random moving object by incorporating a simple visual system. Thus, simple tonic inputs alone can be used to modulate the speed and direction of the swimming robot. (For an excellent paper on the design and comparison of eel-like robots such as Ayers's Undulatory Underwater Robot, Hirose's land-based snake robot, MIT's RoboTuna and RoboPike, and the REEL, refer to [Knutsen, 2000].)

Another possible application is to evolve swimming controllers for sperm-like micro-machines. Due to the small size and power source problems of this kind of robot, it is necessary to find a simple control mechanism so that the controllers can be embedded into these machines to allow efficient swimming.

[2]McIsaac and Ostrowski [1999] modeled the robot as a serial chain of identical planar links and assumed that water is stationary. The water force equations used in their simulations are the same as the ones proposed by Ekeberg which are subsequently used by Ijspeert and me.

8.3 Directions for future research

The current work can be extended in three main directions, as follows.

In biology, it would be a good idea to collaborate with neurobiologists to conduct comprehensive simulations and real biological experiments on intersegmental couplings. Methods such as those presented in [Ayers et al., 1983, Miller and Sigvardt, 2000], Subsections 6.4.1 and 7.4.2 could be used to guide the types of experiments which might lead to interesting results. Experiments focusing on how the swimming performance and the corresponding neural networks change throughout development could provide a good understanding of the adaptability of real biological controllers. This in turn could help us to design adaptive robot controllers. Also, experiments relating neural organization to swimming efficiency could confirm whether or not the evolved efficient controller is biologically plausible. If it is, the fitness function used to evolve this controller would be shown to have captured the essential features of efficient swimming.

In ALife, Ijspeert [2001] extended his lamprey locomotion controller for the swimming and trotting of a salamander in 3D computer simulation. It would be interesting to extend the fitness function for evolving efficient controllers to generate controllers which allow the model salamander to move efficiently (both on land and underwater), and adapt to changes in body parameters as well as being robust against noise in neural connections. The effect of sensory feedback on visual tracking as well as the effect of vestibular compensation on swimming motion could be studied. The 3D simulation software MathEngine (now called Vortex) provided by MathEngine plc (http://www.mathengine.com) running on a Beowulf cluster (http://www.beowulf.org) would provide a good experimental platform for this work.

In robotics, the most interesting application would be to test the biological controller, the efficient controller and a sinusoidal controller on a future 10 link version of the REEL. Some of the possible experiments are:

- Comparison of swimming speed.
- Comparison of swimming efficiency. As the evolved efficient controller can swim with the same efficiency as the real lamprey, they could be the ideal controllers for the REEL given that its battery power is limited. Efficient swimming is therefore necessary for the robot to move from place to place. It would therefore be interesting to compare the distance which the robot can travel when it is embedded with different controllers but with the same amount of energy.

The next challenge to be overcome in the development of a complete robotic lamprey is the integration of sensory feedback (such as visual, tactile and vestibular compensation) to the robot with one extra DOF at each joint. The complete animat can then be used to investigate visual tracking and speed barrier crossing in a real world environment. The results could then be compared with those obtained through computer simulations to give further understanding of animal and robotic motor control.

Appendix A

Neural configuration and swimming efficiency (prototype controllers)

	MNl	EINl	LINl	CINl	CINr	LINr	EINr	MNr	BS
MNl	-	1.0 [5, 5]	-	-	-2.0 [5, 5]	-			5.0
EINl	-	0.4 [2, 2]	-	-	-2.0 [1, 10]	-	-	-	2.0
LINl	-	13.0 [5, 5]	-	-	-1.0 [1, 10]	-	-		5.0
CINl	-	3.0 [2, 2]	-1.0 [5, 5]	-	-2.0 [1, 10]	-	-	-	7.0
CINr	-		-	-2.0 [1, 10]	-	-1.0 [5, 5]	3.0 [2, 2]	-	7.0
LINr	-	-	-	-1.0 [1, 10]	-	-	13.0 [5, 5]	-	5.0
EINr	-	-	-	-2.0 [1, 10]	-	-	0.4 [2, 2]	-	2.0
MNr	-	-	-	-2.0 [5, 5]	-	-	1.0 [5, 5]	-	5.0

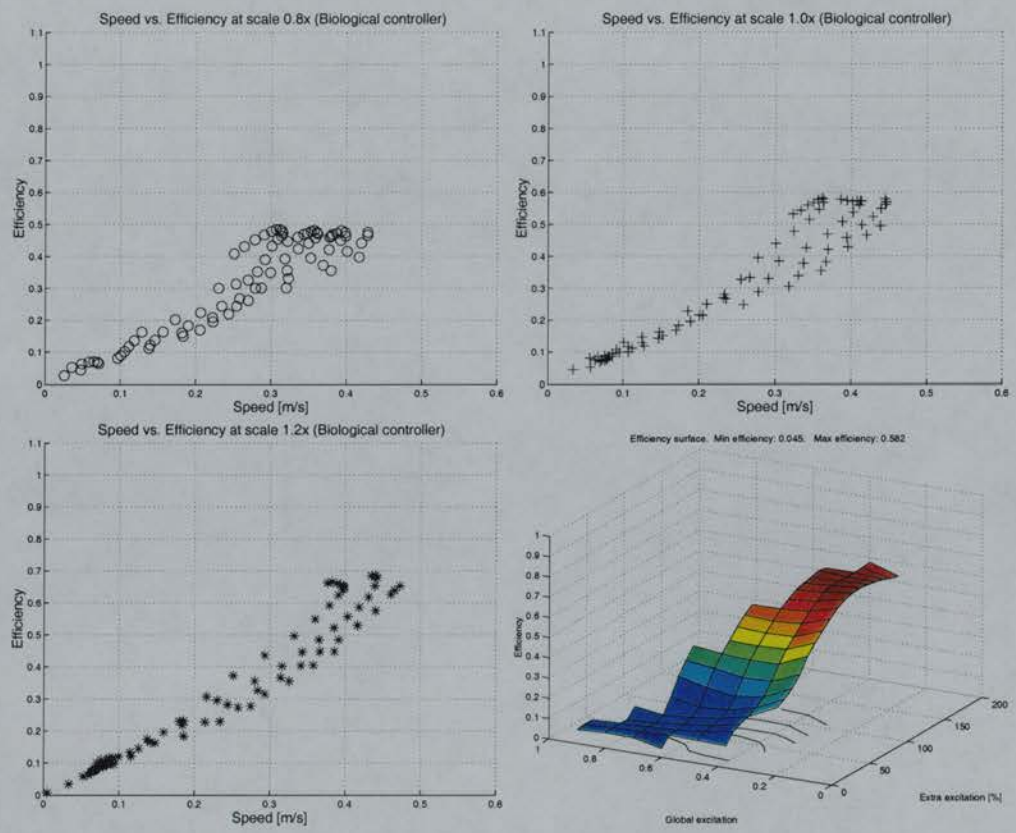


Figure A.1: Neural configuration and swimming efficiency of the biological controller.

	MNl	EINl	LINl	CINl	CINr	LINr	EINr	MNr	BS
MNl	-	1.0 [4, 1]	-	-	-2.0 [11, 11]	-			5.0
EINl	-	0.4 [5, 3]	-	-	-2.0 [1, 0]	-	-	-	2.0
LINl	-	13.0 [4, 1]	-	-	-1.0 [9, 4]	-	-		5.0
CINl	-	3.0 [11, 2]	-1.0 [3, 9]	-	-2.0 [0, 5]	-	-	-	7.0
CINr	-		-	-2.0 [0, 5]	-	-1.0 [3, 9]	3.0 [11, 2]	-	7.0
LINr	-	-	-	-1.0 [9, 4]	-	-	13.0 [4, 1]	-	5.0
EINr	-	-	-	-2.0 [1, 0]	-	-	0.4 [5, 3]	-	2.0
MNr	-	-	-	-2.0 [11, 11]	-	-	1.0 [4, 1]	-	5.0

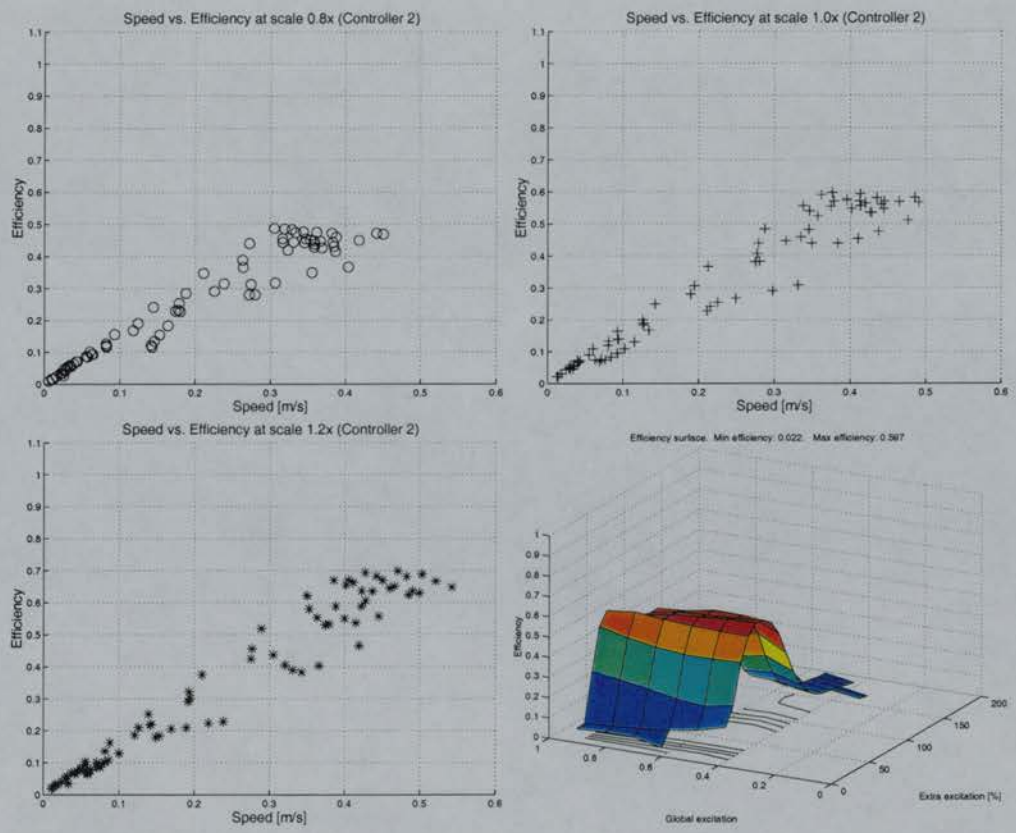


Figure A.2: Neural configuration and swimming efficiency of the controller 2.

	MNl	EINl	LINl	CINl	CINr	LINr	EINr	MNr	BS
MNl	-	-0.4 [9, 2]	-	-3.2 [8, 1]	-	-	-	-	3.8
EINl	-	-0.8 [12, 4]	-	-3.8 [12, 10]	-0.7 [1, 10]	-	-0.9 [5, 10]	-	0.8
LINl	-	-	-	-	-	-	-	-	0.0
CINl	-	-	-	-	-3.7 [9, 9]	-	-3.5 [2, 2]	-	13.6
CINr	-	-3.5 [2, 2]	-	-3.7 [9, 9]	-	-	-	-	13.6
LINr	-	-	-	-	-	-		-	0.0
EINr	-	-0.9 [5, 10]	-	-0.7 [1, 10]	-3.8 [12, 10]	-	-0.8 [12, 4]	-	0.8
MNr	-	-	-	-	-3.2 [8, 1]	-	-0.4 [9, 2]	-	3.8

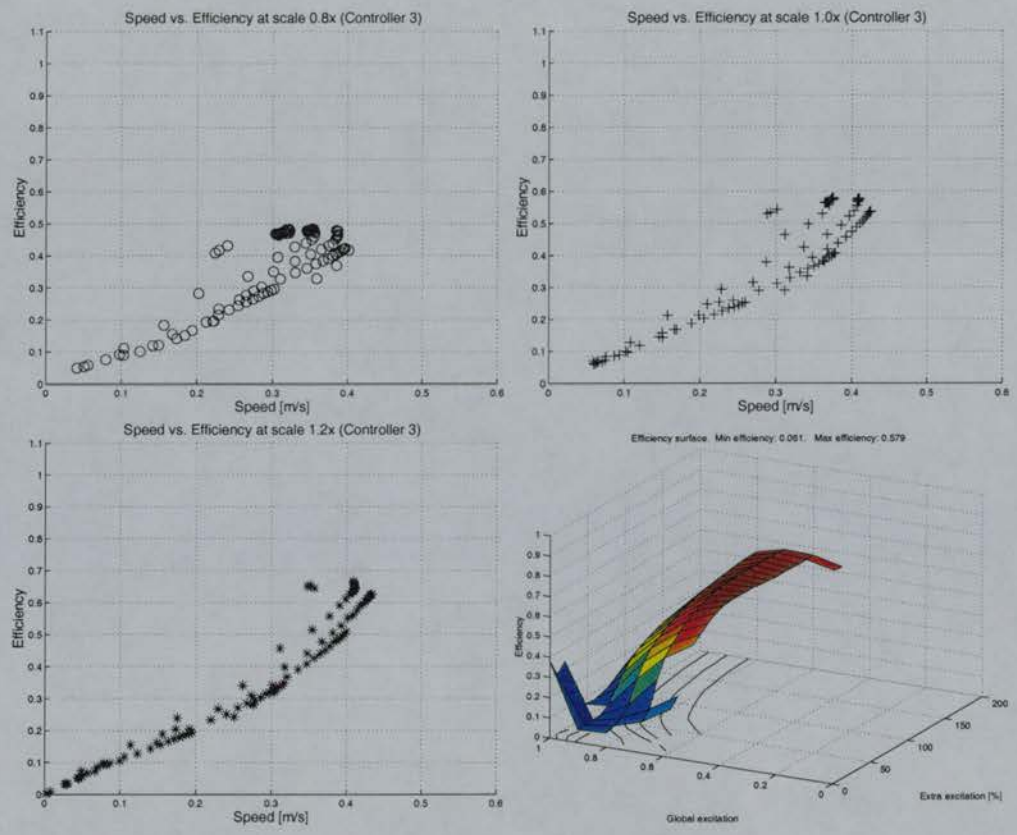


Figure A.3: Neural configuration and swimming efficiency of the controller 3.

	MNl	EINl	LINl	CINl	CINr	LINr	EINr	MNr	BS
MNl	-	-	-	-	-2.5 [0, 5]	-	-	-	5.5
EINl	-	-	-4.2 [8, 4]	-	-2.5 [8, 10]	-	-	-	0.9
LINl	-	14.6 [12, 11]	-	-	-	-0.6 [4, 6]	-	-	4.8
CINl	-	-	-4.8 [7, 0]	-	-1.3 [1, 5]	-	-	-	7.8
CINr	-	-	-	-1.3 [1, 5]	-	-4.8 [7, 0]	-	-	7.8
LINr	-	-	-0.6 [4, 6]	-	-	-	14.6 [12, 11]	-	4.8
EINr	-	-	-	-2.5 [8, 10]	-	-4.2 [8, 4]	-	-	0.9
MNr	-	-	-	-2.5 [0, 5]	-	-	-	-	5.5

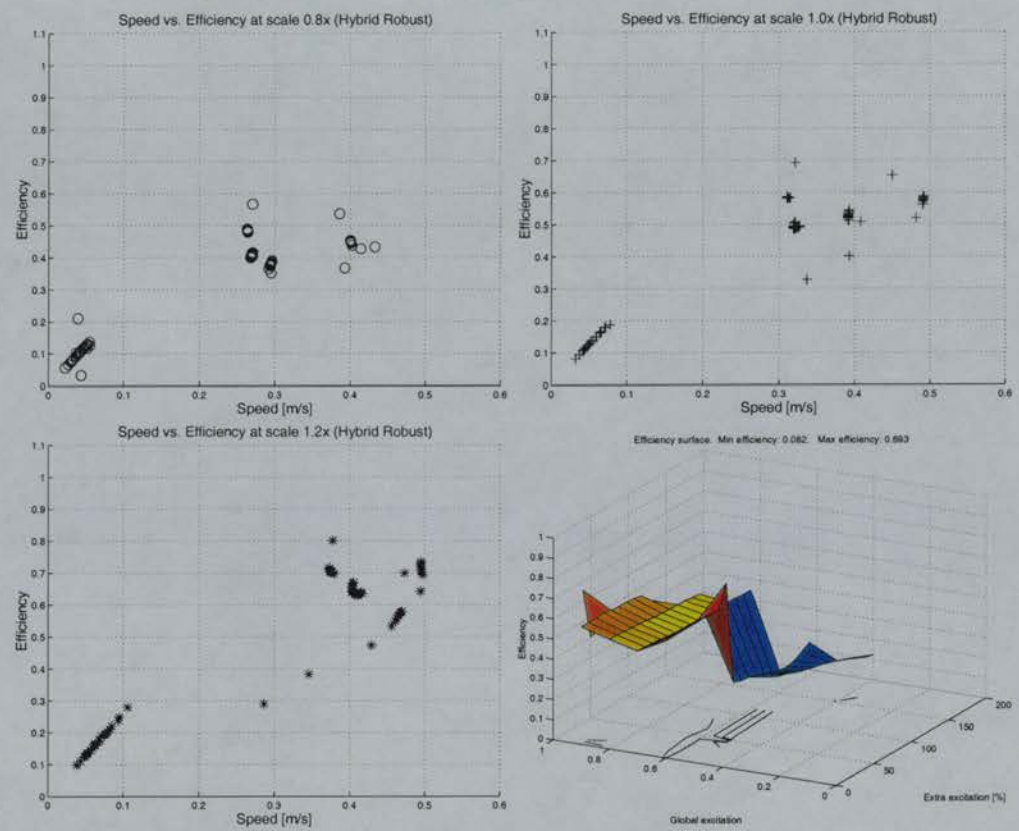


Figure A.4: Neural configuration and swimming efficiency of the hybrid robust controller.

Appendix B

Neural configuration and performance surfaces of controllers evolved through the bigmax approach

	MNl	EINl	LINl	CINl	CINr	LINr	EINr	MNr	BS
MNl	-	1.0 [5,3]	-	-	-2.0 [3,2]	-			5.0
EINl	-	0.4 [2,11]	-	-	-2.0 [3,5]	-	-	-	2.0
LINl	-	13.0 [8,4]	-	-	-1.0 [4,2]	-	-		5.0
CINl	-	3.0 [0,2]	-1.0 [0,1]	-	-2.0 [0, 7]	-	-	-	7.0
CINr	-		-	-2.0 [0 ,7]	-	-1.0 [0,1]	3.0 [0, 2]	-	7.0
LINr	-	-	-	-1.0 [4, 2]	-	-	13.0 [8,4]	-	5.0
EINr	-	-	-	-2.0 [3,5]	-	-	0.4 [2,11]	-	2.0
MNr	-	-	-	-2.0 [3,2]	-	-	1.0 [5,3]	-	5.0

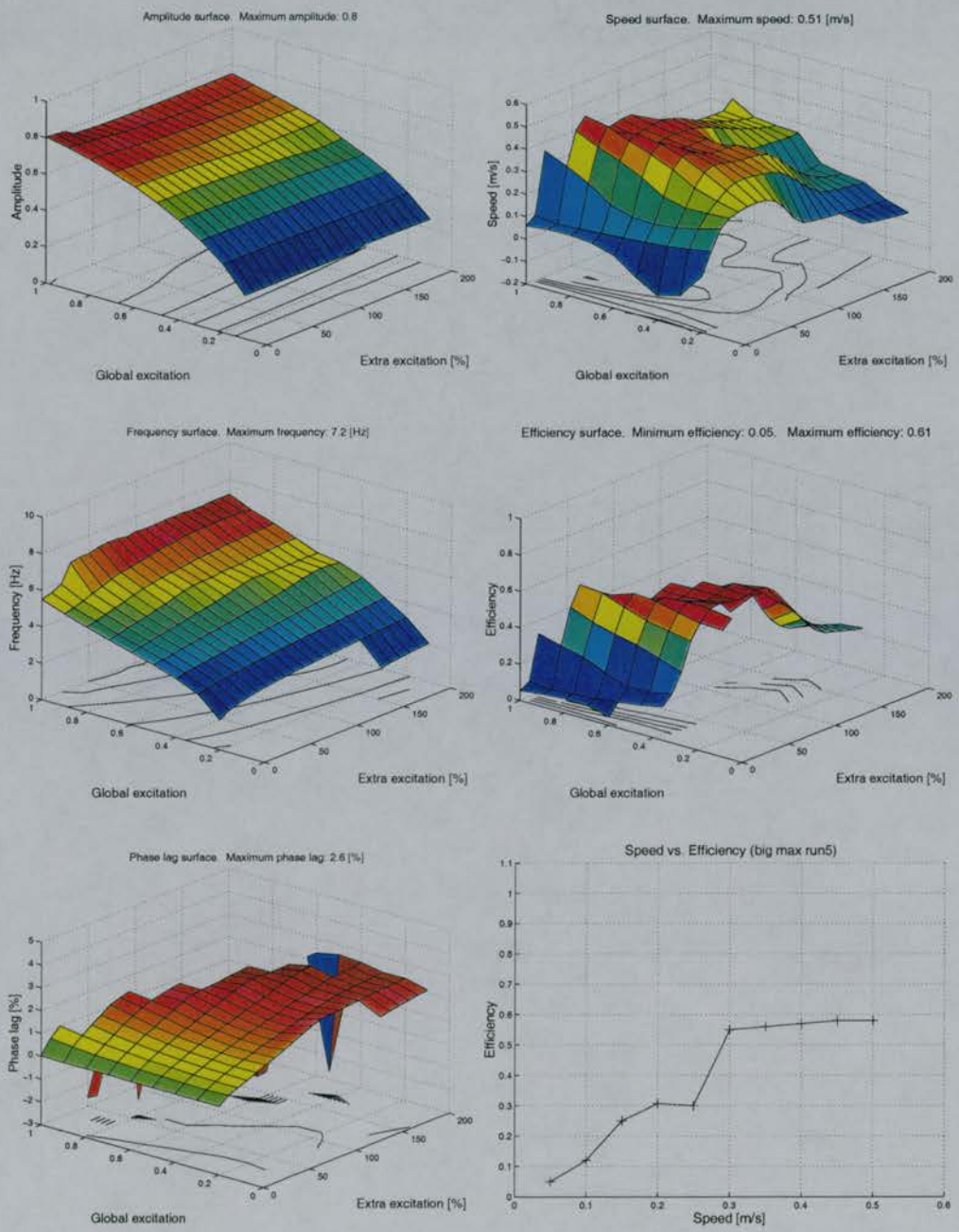


Figure B.1: Neural configuration and performance surface of run5 controller.

	MNl	EINl	LINl	CINl	CINr	LINr	EINr	MNr	BS
MNl	-	1.0 [0,0]	-	-	-2.0 [9,1]	-	-	-	5.0
EINl	-	0.4 [7, 6]	-	-	-2.0 [3, 3]	-	-	-	2.0
LINl	-	13.0 [7,0]	-	-	-1.0 [5,5]	-	-	-	5.0
CINl	-	3.0 [7,2]	-1.0 [3,4]	-	-2.0 [0, 6]	-	-	-	7.0
CINr	-	-	-	-2.0 [0 ,6]	-	-1.0 [3,4]	3.0 [7, 2]	-	7.0
LINr	-	-	-	-1.0 [5, 5]	-	-	13.0 [7, 0]	-	5.0
EINr	-	-	-	-2.0 [3,3]	-	-	0.4 [7, 6]	-	2.0
MNr	-	-	-	-2.0 [9, 1]	-	-	1.0 [0,0]	-	5.0

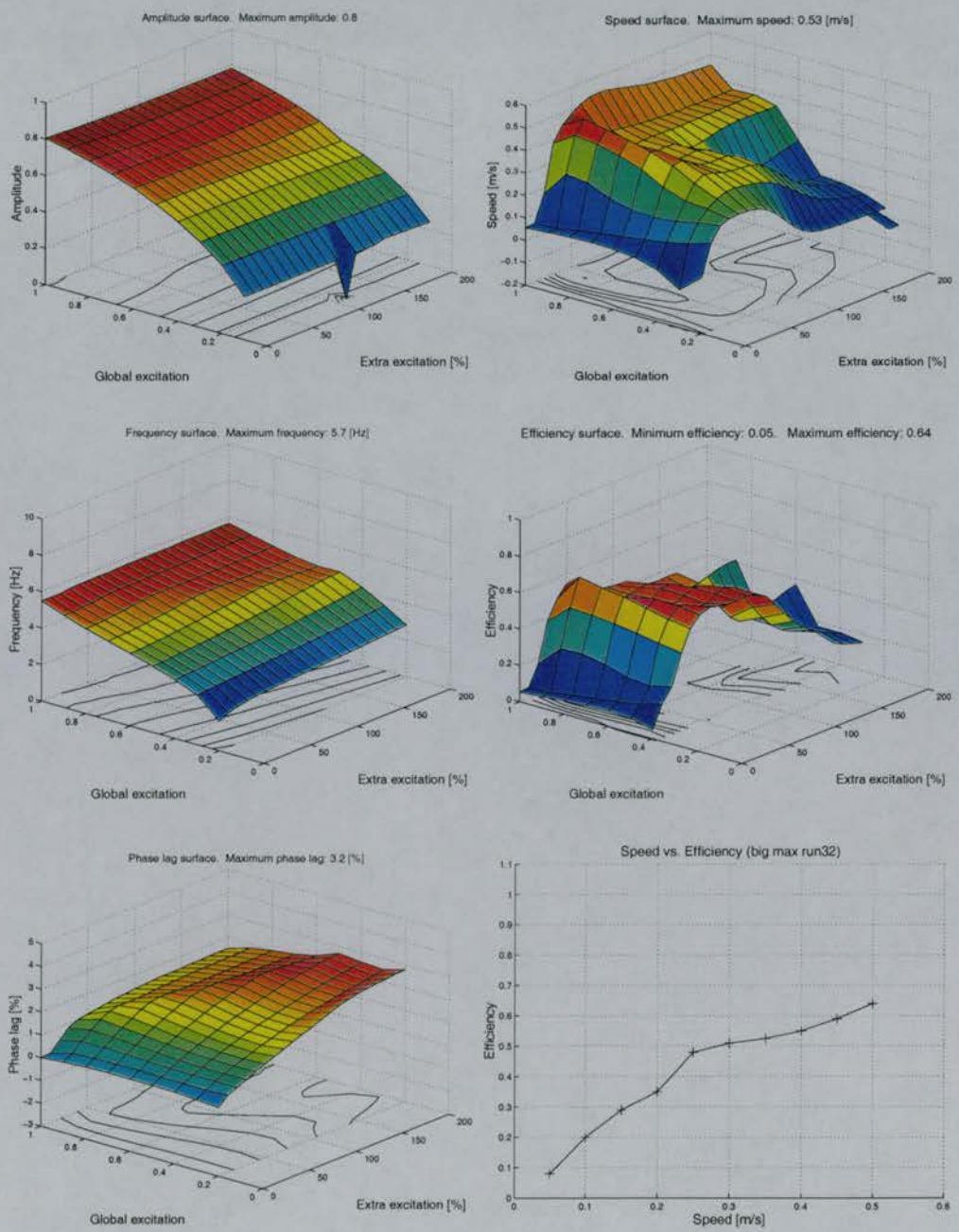


Figure B.2: Neural configuration and performance surface of run32 controller.

	MNl	EINl	LINl	CINl	CINr	LINr	EINr	MNr	BS
MNl	-	1.0 [2,12]	-	-	-2.0 [4,3]	-	-	-	5.0
EINl	-	0.4 [8,10]	-	-	-2.0 [5, 6]	-	-	-	2.0
LINl	-	13.0 [6,3]	-	-	-1.0 [4,0]	-	-	-	5.0
CINl	-	3.0 [3,1]	-1.0 [0,2]	-	-2.0 [0, 7]	-	-	-	7.0
CINr	-	-	-	-2.0 [0 ,7]	-	-1.0 [0,2]	3.0 [3, 1]	-	7.0
LINr	-	-	-	-1.0 [4, 0]	-	-	13.0 [6, 3]	-	5.0
EINr	-	-	-	-2.0 [5,6]	-	-	0.4 [8,10]	-	2.0
MNr	-	-	-	-2.0 [4, 3]	-	-	1.0 [2,12]	-	5.0

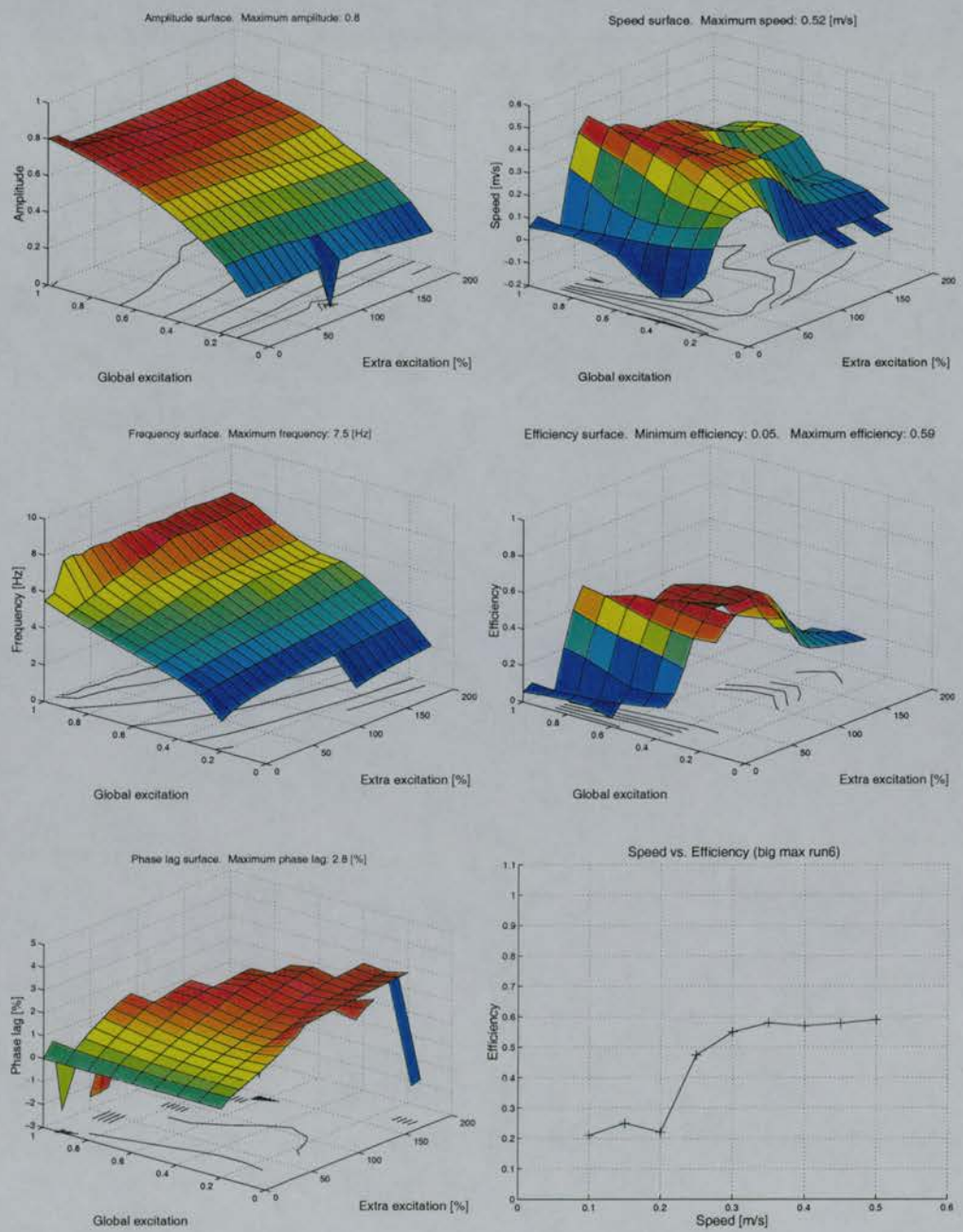


Figure B.3: Neural configuration and performance surface of run6 controller.

	MNl	ElNl	LINl	ClNl	ClNr	LINr	ElNr	MNr	BS
MNl	-	1.0 [10,7]	-	-	-2.0 [7,10]	-	-	-	5.0
ElNl	-	0.4 [9, 5]	-	-	-2.0 [6, 3]	-	-	-	2.0
LINl	-	13.0 [6,0]	-	-	-1.0 [5,12]	-	-	-	5.0
ClNl	-	3.0 [7,5]	-1.0 [1,5]	-	-2.0 [3, 8]	-	-	-	7.0
ClNr	-	-	-	-2.0 [3 ,8]	-	-1.0 [1,5]	3.0 [7, 5]	-	7.0
LINr	-	-	-	-1.0 [5, 12]	-	-	13.0 [6, 0]	-	5.0
ElNr	-	-	-	-2.0 [6,3]	-	-	0.4 [9, 5]	-	2.0
MNr	-	-	-	-2.0 [7,10]	-	-	1.0 [10,7]	-	5.0

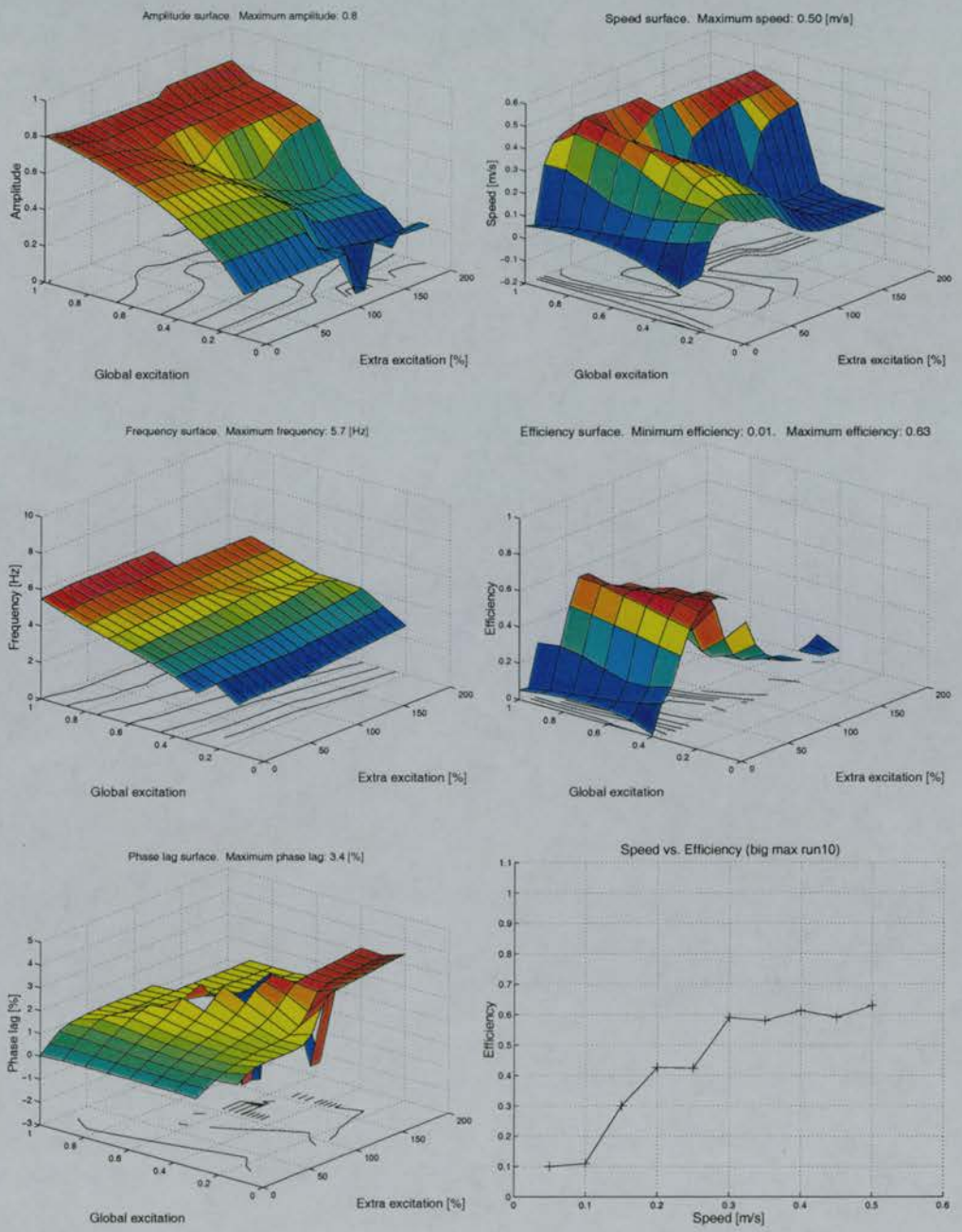


Figure B.4: Neural configuration and performance surface of run10 controller.

	MNl	EINl	LINl	CINl	CINr	LINr	EINr	MNr	BS
MNl	-	-0.4 [12,3]	-	-3.2 [5, 12]	-	-	-	-	3.8
EINl	-	-0.8 [1, 9]	-	-3.8 [8, 6]	-0.7 [12, 11]	-	-0.9 [2, 7]	-	0.8
LINl	-	-	-	-	-	-	-	-	0.0
CINl	-	-	-	-	-3.7 [4, 6]	-	-3.5 [4, 8]	-	13.6
CINr	-	-3.5 [4, 8]	-	-3.7 [4 ,6]	-	-	-	-	13.6
LINr	-	-	-	-	-	-		-	0.0
EINr	-	-0.9 [2, 7]	-	-0.7 [12, 11]	-3.8 [8, 6]	-	-0.8 [1, 9]	-	0.8
MNr	-	-	-	-	-3.2 [5, 12]	-	-0.4 [12, 3]	-	3.8

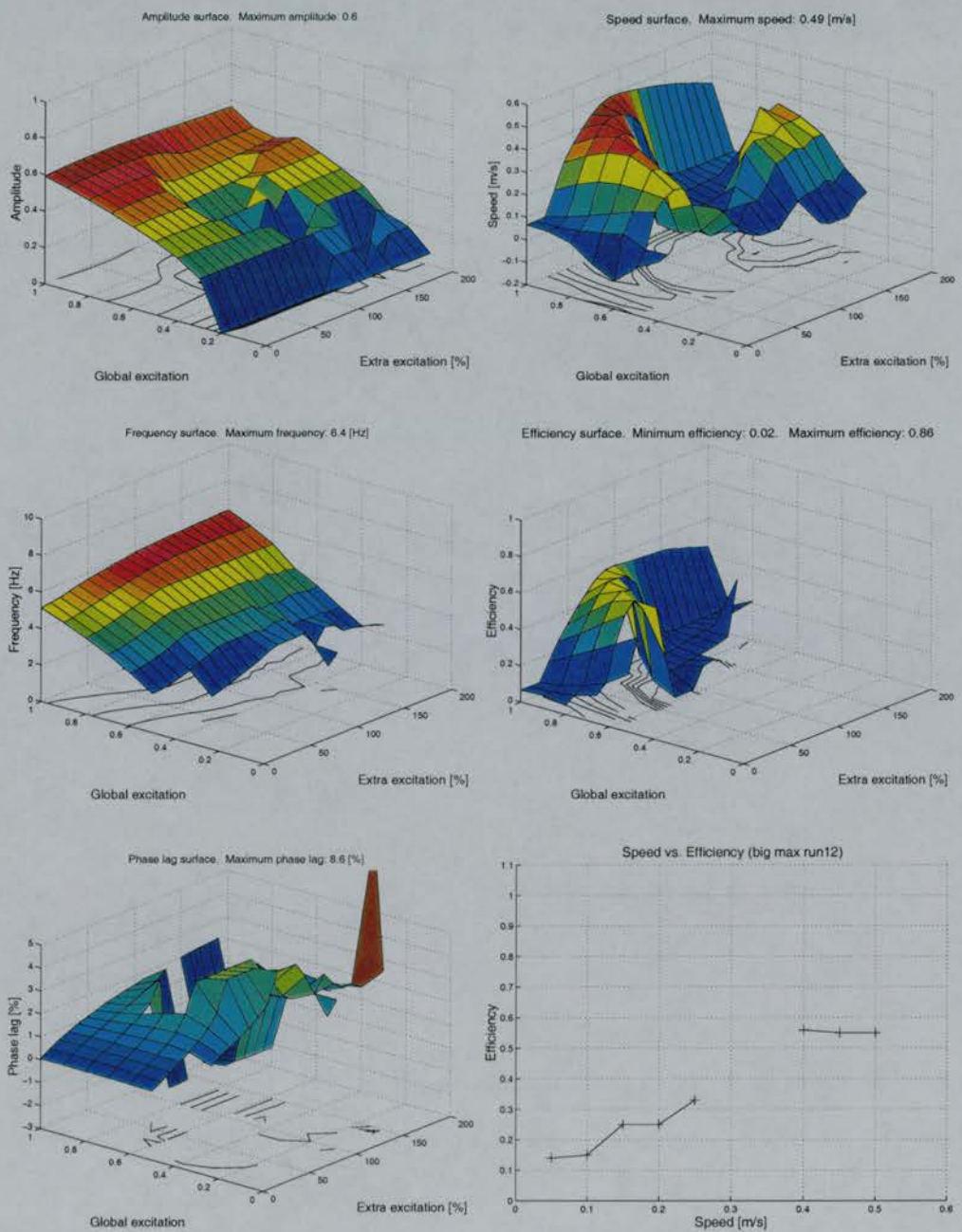


Figure B.5: Neural configuration and performance surface of run12 controller.

	MNl	EINl	LINl	CINl	CINr	LINr	EINr	MNr	BS
MNl	-	-0.4 [8 ,1]	-	-3.2 [5, 9]	-	-	-	-	3.8
EINl	-	-0.8 [2, 2]	-	-3.8 [5, 3]	-0.7 [5 , 10]	-	-0.9 [1, 6]	-	0.8
LINl	-	-	-	-	-	-	-	-	0.0
CINl	-	-	-	-	-3.7 [8, 3]	-	-3.5 [8, 10]	-	13.6
CINr	-	-3.5 [8,10]	-	-3.7 [8 ,3]	-	-	-	-	13.6
LINr	-	-	-	-	-	-		-	0.0
EINr	-	-0.9 [1, 6]	-	-0.7 [5 , 10]	-3.8 [5, 3]	-	-0.8 [2, 2]	-	0.8
MNr	-	-	-	-	-3.2 [5, 9]	-	-0.4 [8, 1]	-	3.8

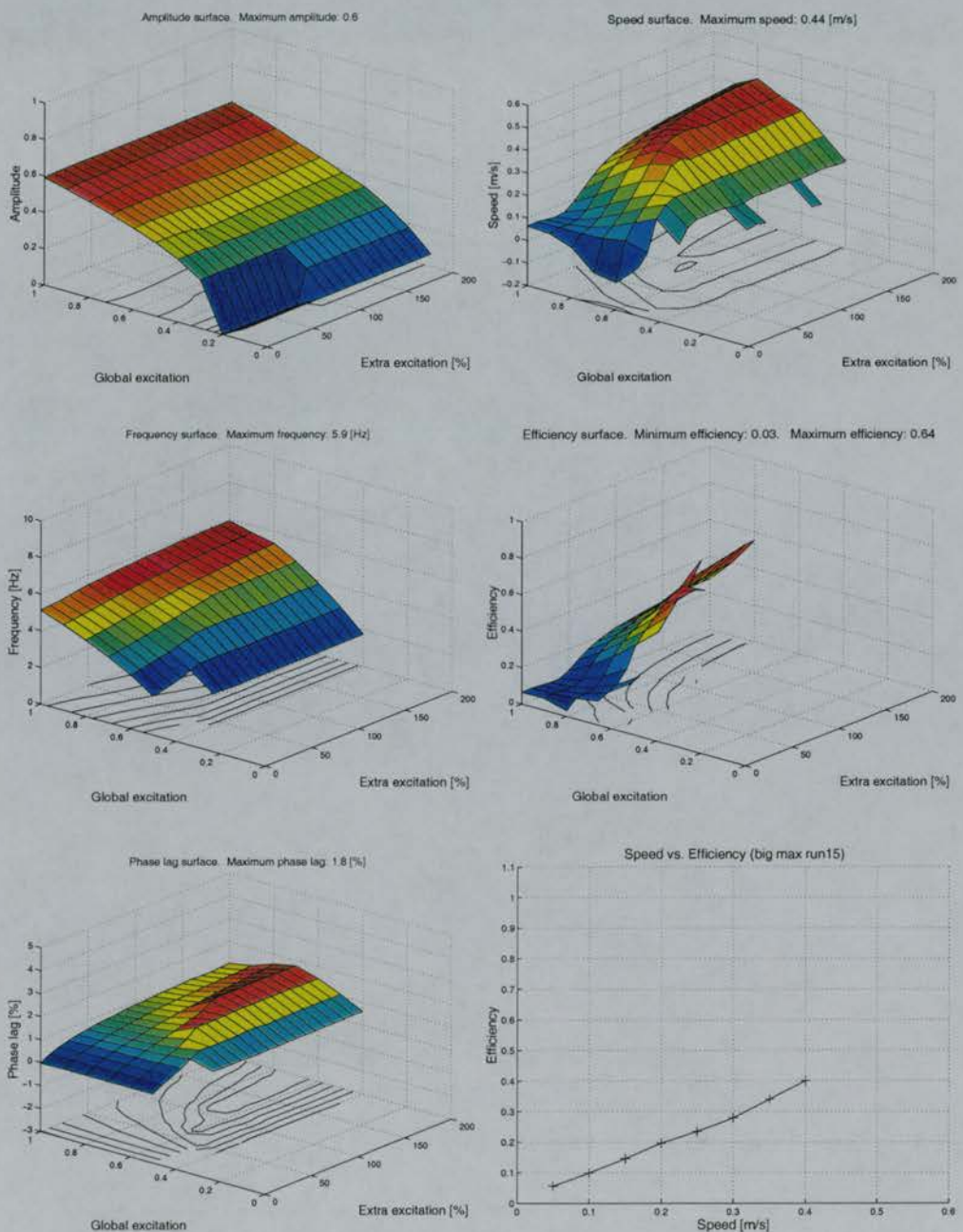


Figure B.6: Neural configuration and performance surface of run15 controller.

	MNl	EINl	LINl	CINl	CINr	LINr	EINr	MNr	BS
MNl	-	-	-	-	-2.5 [12,7]	-	-	-	5.5
EINl	-	-	-4.2 [8,9]	-	-2.5 [9 ,8]	-	-	-	0.9
LINl	-	14.6 [9,7]	-	-	-	-0.6 [7, 3]	-	-	4.8
CINl	-	-	-4.8 [7,3]	-	-1.3 [3, 5]	-	-	-	7.8
CINr	-	-	-	-1.3 [3,5]	-	-4.8 [7, 3]	-	-	7.8
LINr	-	-	-0.6 [7,3]	-	-	-	14.6 [9,7]	-	4.8
EINr	-	-	-	-2.5 [9 , 8]	-	-4.2 [8, 9]	-	-	0.9
MNr	-	-	-	-2.5 [12, 7]	-	-	-	-	5.5

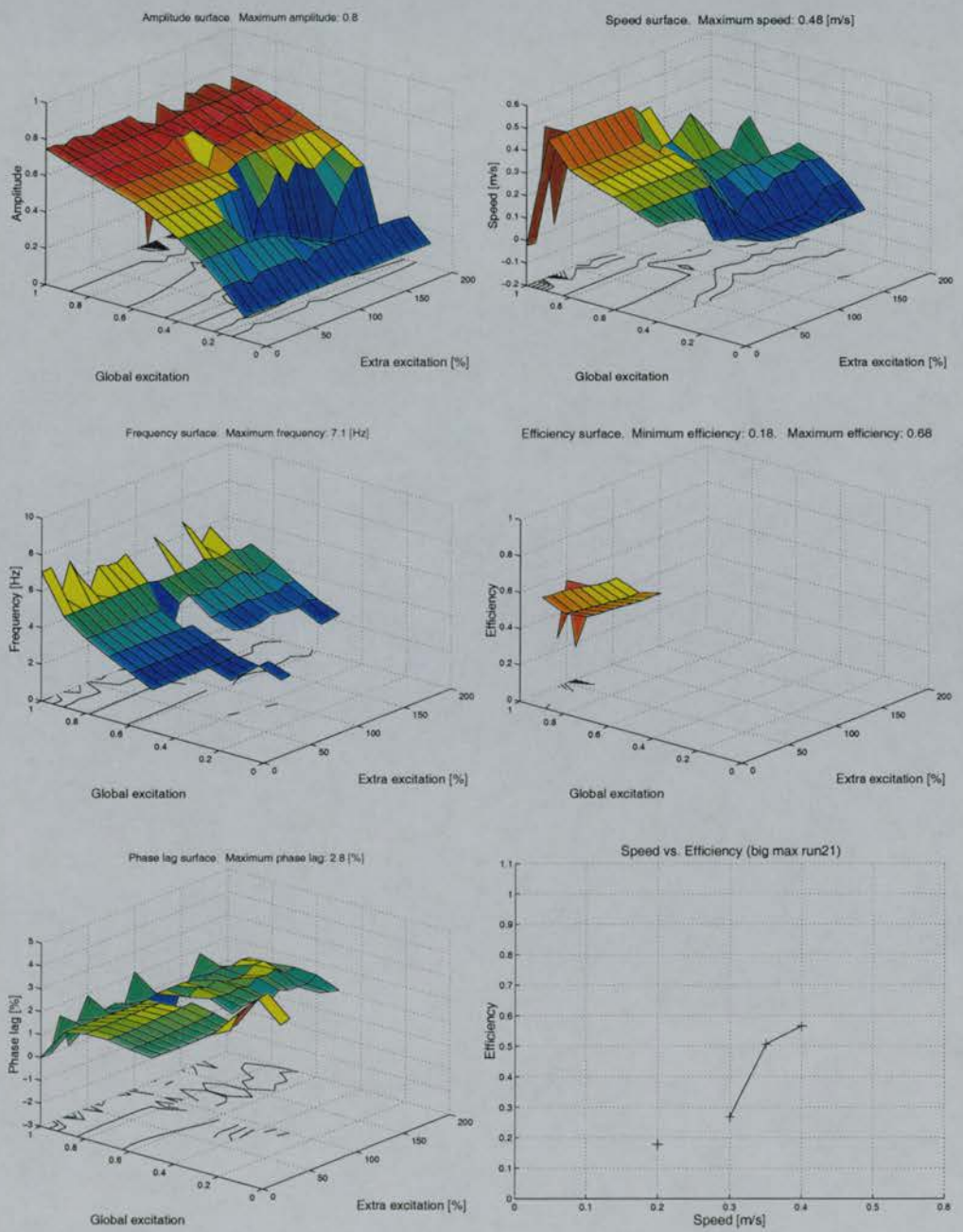


Figure B.7: Neural configuration and performance surface of run21 controller.

	MNl	EINl	LINl	CINl	CINr	LINr	EINr	MNr	BS
MNl	-	-	-	-	-2.5 [10,6]	-	-	-	5.5
EINl	-	-	-4.2 [9,9]	-	-2.5 [10,7]	-	-	-	0.9
LINl	-	14.6 [9,6]	-	-	-	-0.6 [8, 4]	-	-	4.8
CINl	-	-	-4.8 [4,3]	-	-1.3 [2, 4]	-	-	-	7.8
CINr	-	-	-	-1.3 [2,4]	-	-4.8 [4, 3]	-	-	7.8
LINr	-	-	-0.6 [8,4]	-	-	-	14.6 [9,6]	-	4.8
EINr	-	-	-	-2.5 [10, 7]	-	-4.2 [9, 9]	-	-	0.9
MNr	-	-	-	-2.5 [10, 6]	-	-	-	-	5.5

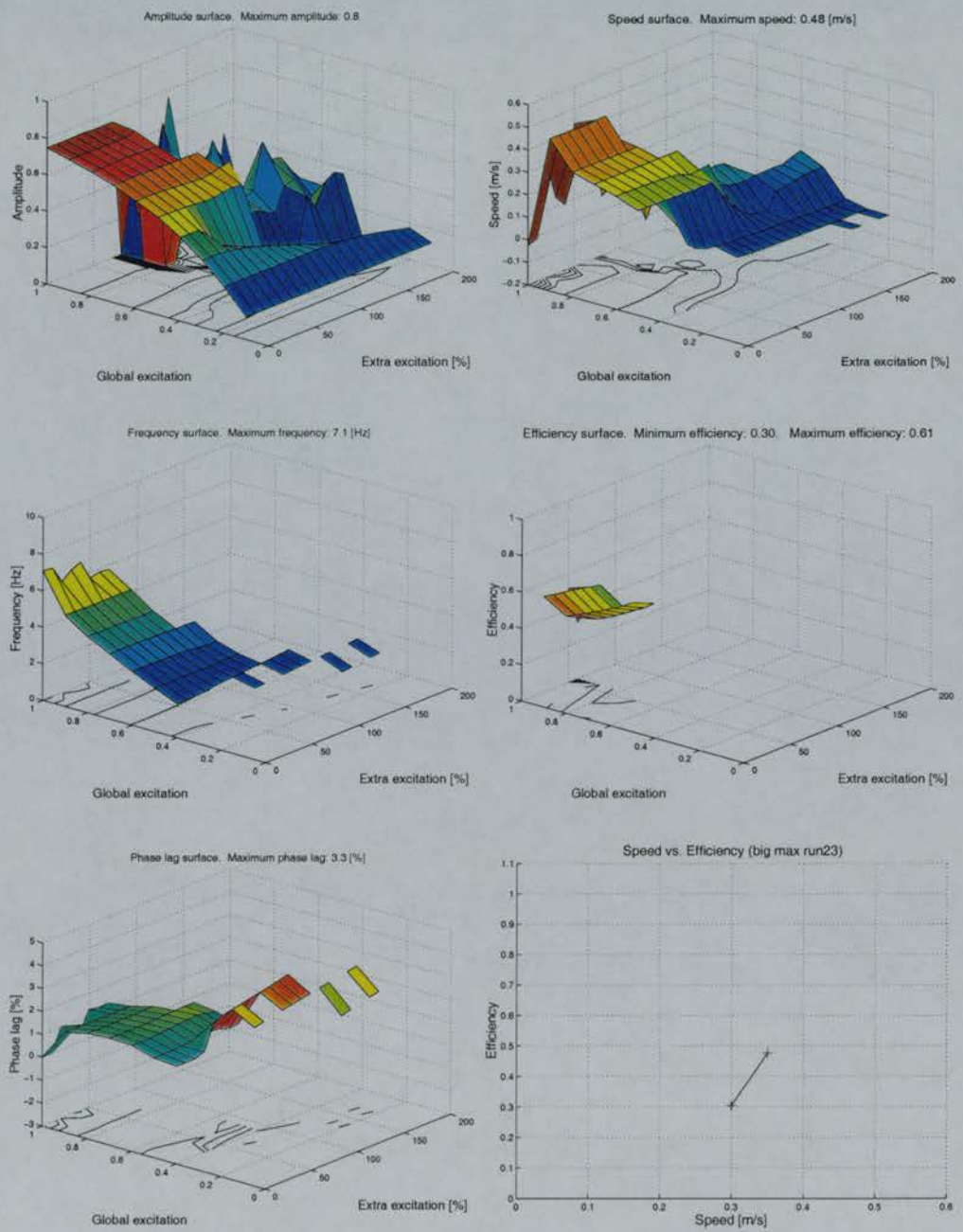


Figure B.8: Neural configuration and performance surface of run23 controller.

	MNl	EINl	LINl	CINl	CINr	LINr	EINr	MNr	BS
MNl	-	-	-	-	-2.5 [8, 4]	-	-	-	5.5
EINl	-	-	-4.2 [2, 10]	-	-2.5 [4, 5]	-	-	-	0.9
LINl	-	14.6 [10, 11]	-	-	-	-0.6 [5, 11]	-	-	4.8
CINl	-	-	-4.8 [12, 9]	-	-1.3 [0, 6]	-	-	-	7.8
CINr	-	-	-	-1.3 [0,6]	-	-4.8 [12, 9]	-	-	7.8
LINr	-	-	-0.6 [5, 11]	-	-	-	14.6 [10, 11]	-	4.8
EINr	-	-	-	-2.5 [4 , 5]	-	-4.2 [2, 10]	-	-	0.9
MNr	-	-	-	-2.5 [8 , 4]	-	-	-	-	5.5

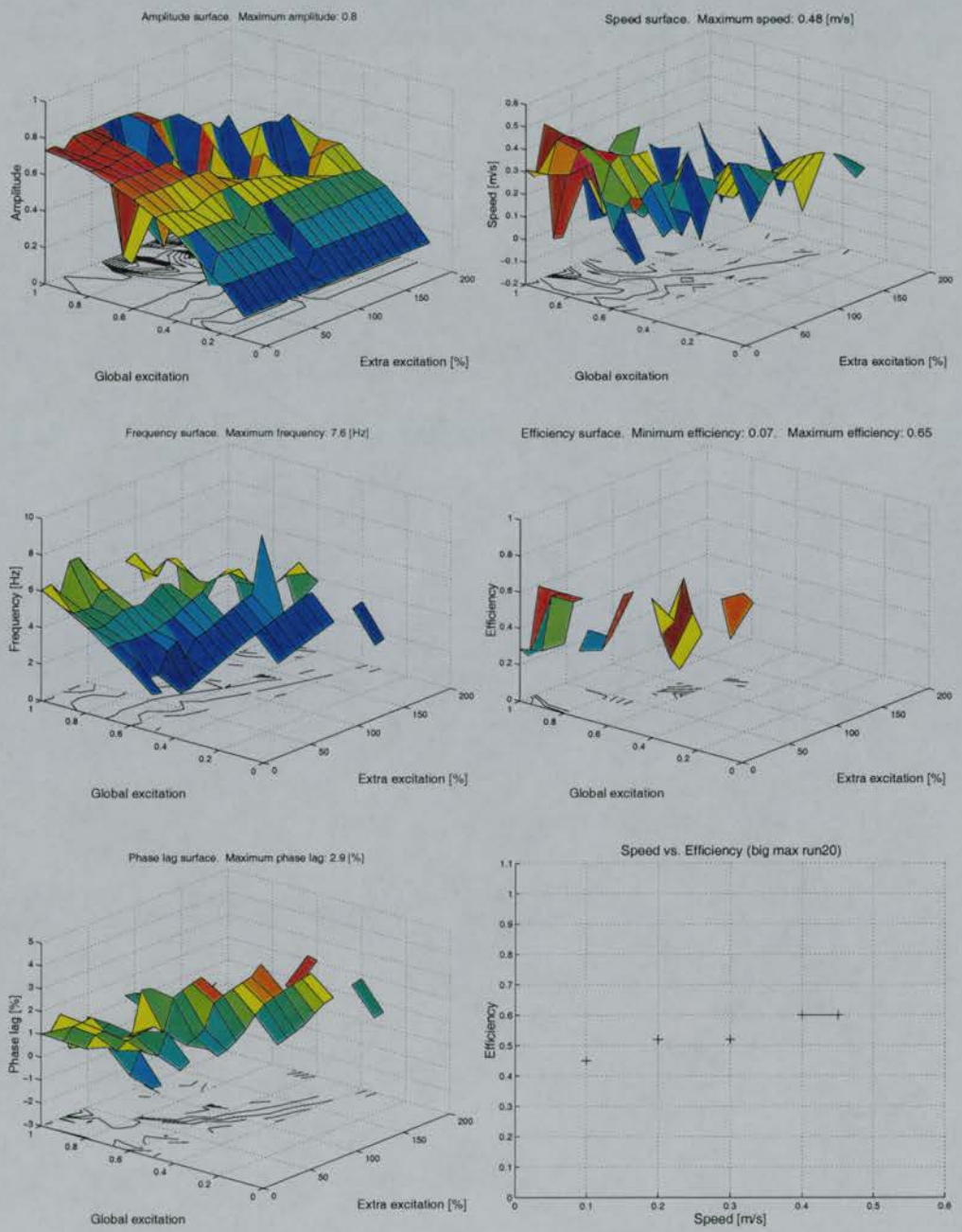


Figure B.9: Neural configuration and performance surface of run20 controller.

	MNl	EINl	LINl	CINl	CINr	LINr	EINr	MNr	BS
MNl	-	-	-	-	-2.5 [8, 4]	-	-	-	5.5
EINl	-	-	-4.2 [7, 11]	-	-2.5 [8, 10]	-	-	-	0.9
LINl	-	14.6 [10,8]	-	-	-	-0.6 [0, 8]	-	-	4.8
CINl	-	-	-4.8 [1 ,1]	-	-1.3 [0, 9]	-	-	-	7.8
CINr	-	-	-	-1.3 [0, 9]	-	-4.8 [1, 1]	-	-	7.8
LINr	-	-	-0.6 [0, 8]	-	-	-	14.6 [10, 8]	-	4.8
EINr	-	-	-	-2.5 [8, 10]	-	-4.2 [7, 11]	-	-	0.9
MNr	-	-	-	-2.5 [8, 4]	-	-	-	-	5.5

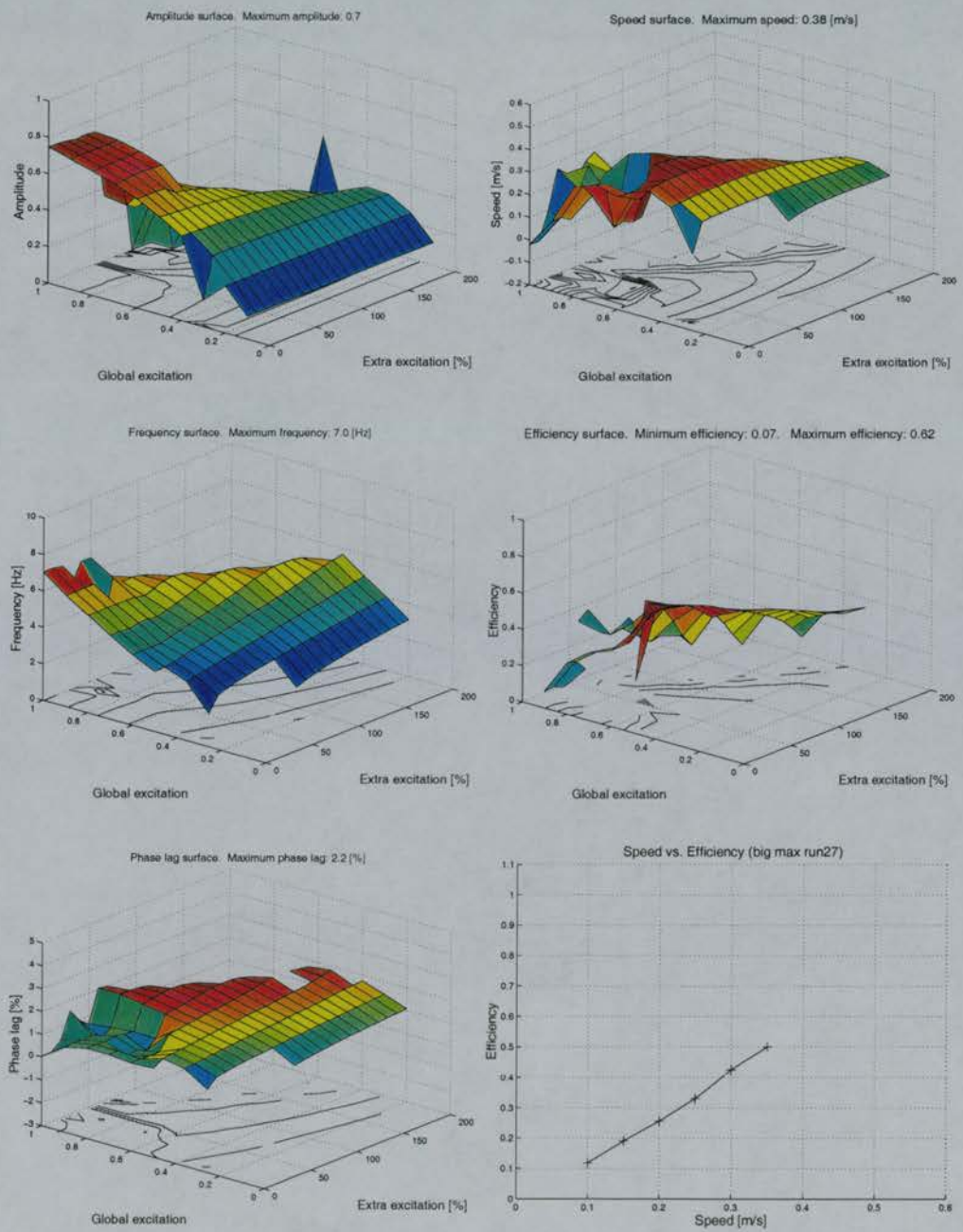


Figure B.10: Neural configuration and performance surface of run27 controller.

Appendix C

Neural configuration and performance surfaces of controllers evolved through the bigmin approach

	MNl	EINl	LINl	CINl	CINr	LINr	EINr	MNr	BS
MNl	-	1.0 [1, 1]	-	-	-2.0 [6, 7]	-			5.0
EINl	-	0.4 [6, 4]	-	-	-2.0 [12, 1]	-	-	-	2.0
LINl	-	13.0 [4, 0]	-	-	-1.0 [0, 7]	-	-		5.0
CINl	-	3.0 [8, 8]	-1.0 [2, 11]	-	-2.0 [1, 7]	-	-	-	7.0
CINr	-		-	-2.0 [1, 7]	-	-1.0 [2, 11]	3.0 [8, 8]	-	7.0
LINr	-	-	-	-1.0 [0, 7]	-	-	13.0 [4, 0]	-	5.0
EINr	-	-	-	-2.0 [12, 1]	-	-	0.4 [6, 4]	-	2.0
MNr	-	-	-	-2.0 [6, 7]	-	-	1.0 [1, 1]	-	5.0

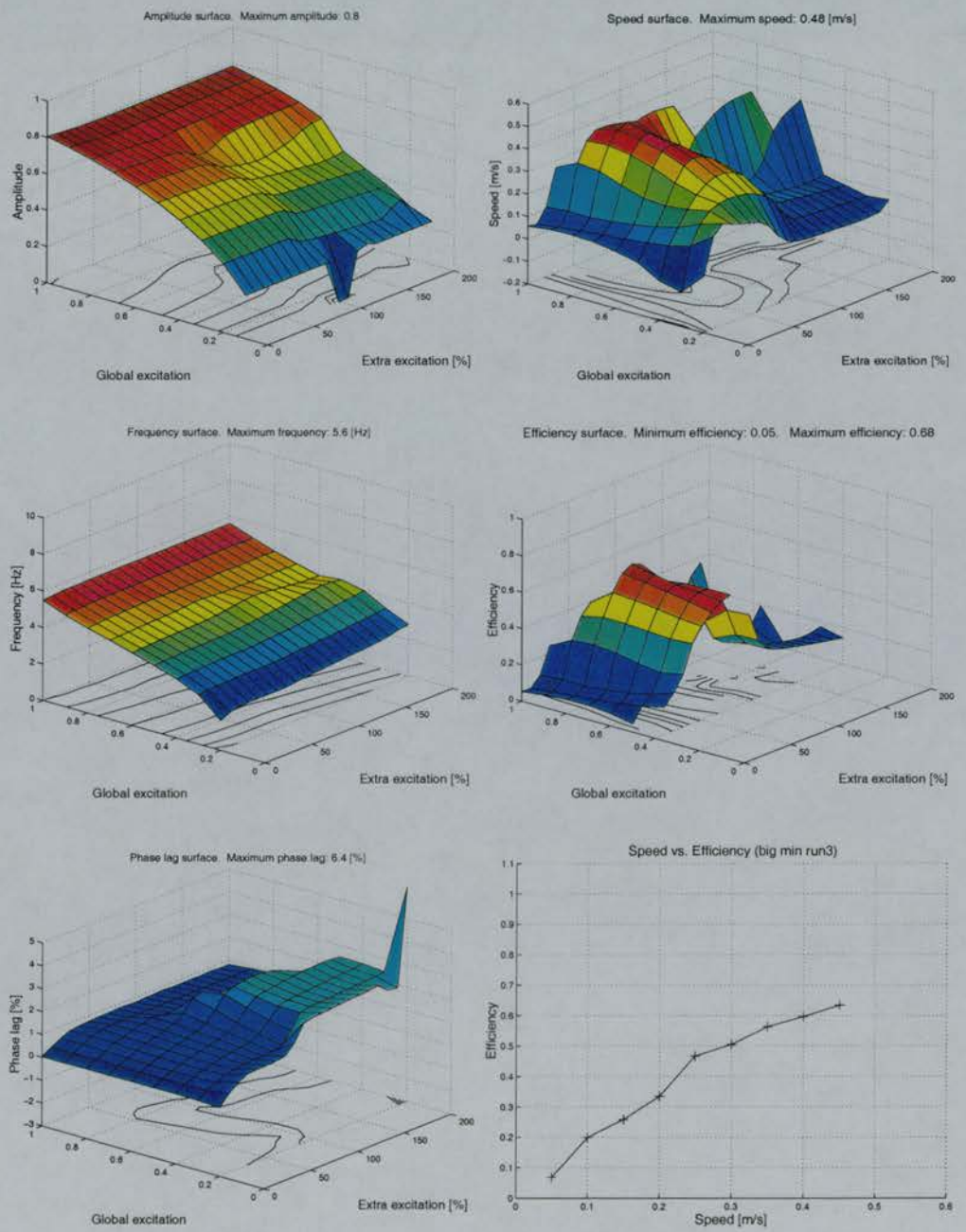


Figure C.1: Neural configuration and performance surface of run3 controller.

	MNl	EINl	LINl	CINl	CINr	LINr	EINr	MNr	BS
MNl	-	1.0 [3, 10]	-	-	-2.0 [9, 2]	-			5.0
EINl	-	0.4 [5, 8]	-	-	-2.0 [1, 5]	-	-	-	2.0
LINl	-	13.0 [1, 7]	-	-	-1.0 [4, 1]	-	-		5.0
CINl	-	3.0 [4, 1]	-1.0 [7, 4]	-	-2.0 [7, 7]	-	-	-	7.0
CINr	-		-	-2.0 [7, 7]	-	-1.0 [7, 4]	3.0 [4, 1]	-	7.0
LINr	-	-	-	-1.0 [4, 1]	-	-	13.0 [1, 7]	-	5.0
EINr	-	-	-	-2.0 [1, 5]	-	-	0.4 [5, 8]	-	2.0
MNr	-	-	-	-2.0 [9, 2]	-	-	1.0 [3, 10]	-	5.0

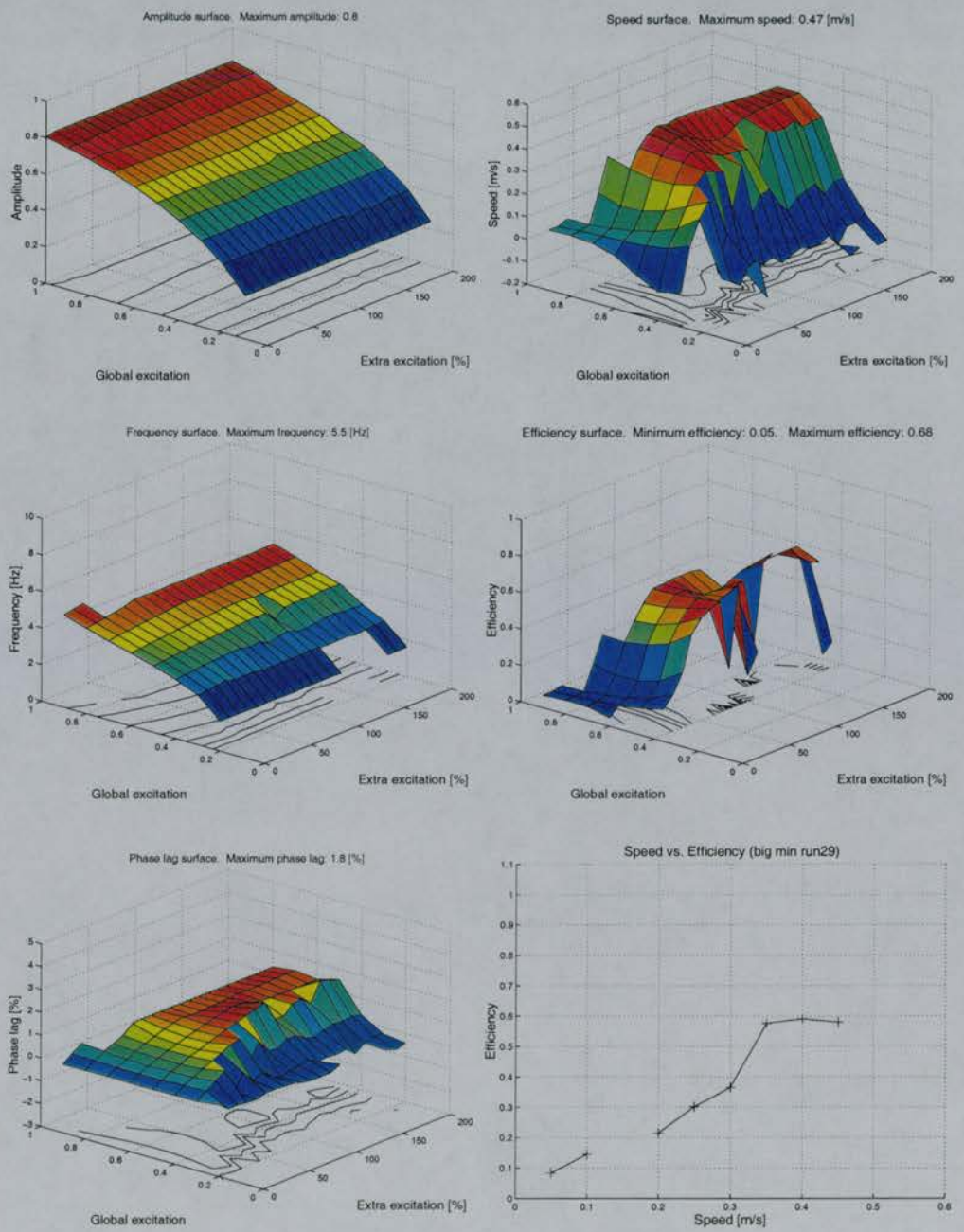


Figure C.2: Neural configuration and performance surface of run29 controller.

	MNl	EINl	LINl	CINl	CINr	LINr	EINr	MNr	BS
MNl	-	1.0 [4, 9]	-	-	-2.0 [4, 4]	-			5.0
EINl	-	0.4 [3, 3]	-	-	-2.0 [3, 5]	-	-	-	2.0
LINl	-	13.0 [4, 2]	-	-	-1.0 [11, 5]	-	-		5.0
CINl	-	3.0 [11,1]	-1.0 [4, 7]	-	-2.0 [5, 7]	-	-	-	7.0
CINr	-		-	-2.0 [5, 7]	-	-1.0 [4, 7]	3.0 [11, 1]	-	7.0
LINr	-	-	-	-1.0 [11, 5]	-	-	13.0 [4, 2]	-	5.0
EINr	-	-	-	-2.0 [5, 5]	-	-	0.4 [3, 3]	-	2.0
MNr	-	-	-	-2.0 [4, 4]	-	-	1.0 [4, 9]	-	5.0

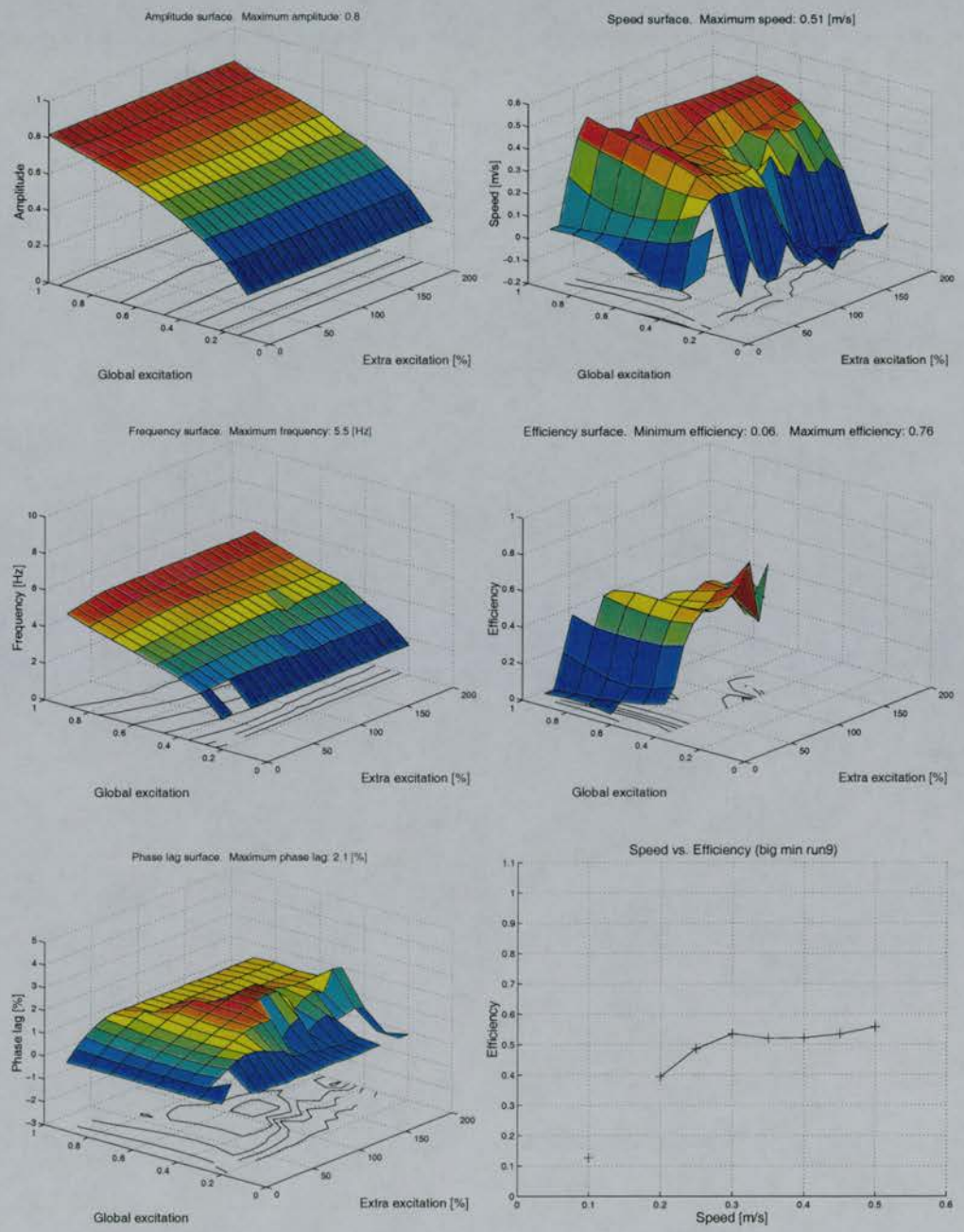


Figure C.3: Neural configuration and performance surface of run9 controller.

	MNl	EINl	LINl	CINl	CINr	LINr	EINr	MNr	BS
MNl	-	1.0 [6, 9]	-	-	-2.0 [7, 7]	-	-	-	5.0
EINl	-	0.4 [11, 2]	-	-	-2.0 [1, 4]	-	-	-	2.0
LINl	-	13.0 [7, 4]	-	-	-1.0 [8, 7]	-	-	-	5.0
CINl	-	3.0 [4, 8]	-1.0 [4, 6]	-	-2.0 [5, 7]	-	-	-	7.0
CINr	-	-	-	-2.0 [5, 7]	-	-1.0 [4, 6]	3.0 [4, 8]	-	7.0
LINr	-	-	-	-1.0 [8, 7]	-	-	13.0 [7, 4]	-	5.0
EINr	-	-	-	-2.0 [1, 4]	-	-	0.4 [11, 2]	-	2.0
MNr	-	-	-	-2.0 [7, 7]	-	-	1.0 [6, 9]	-	5.0

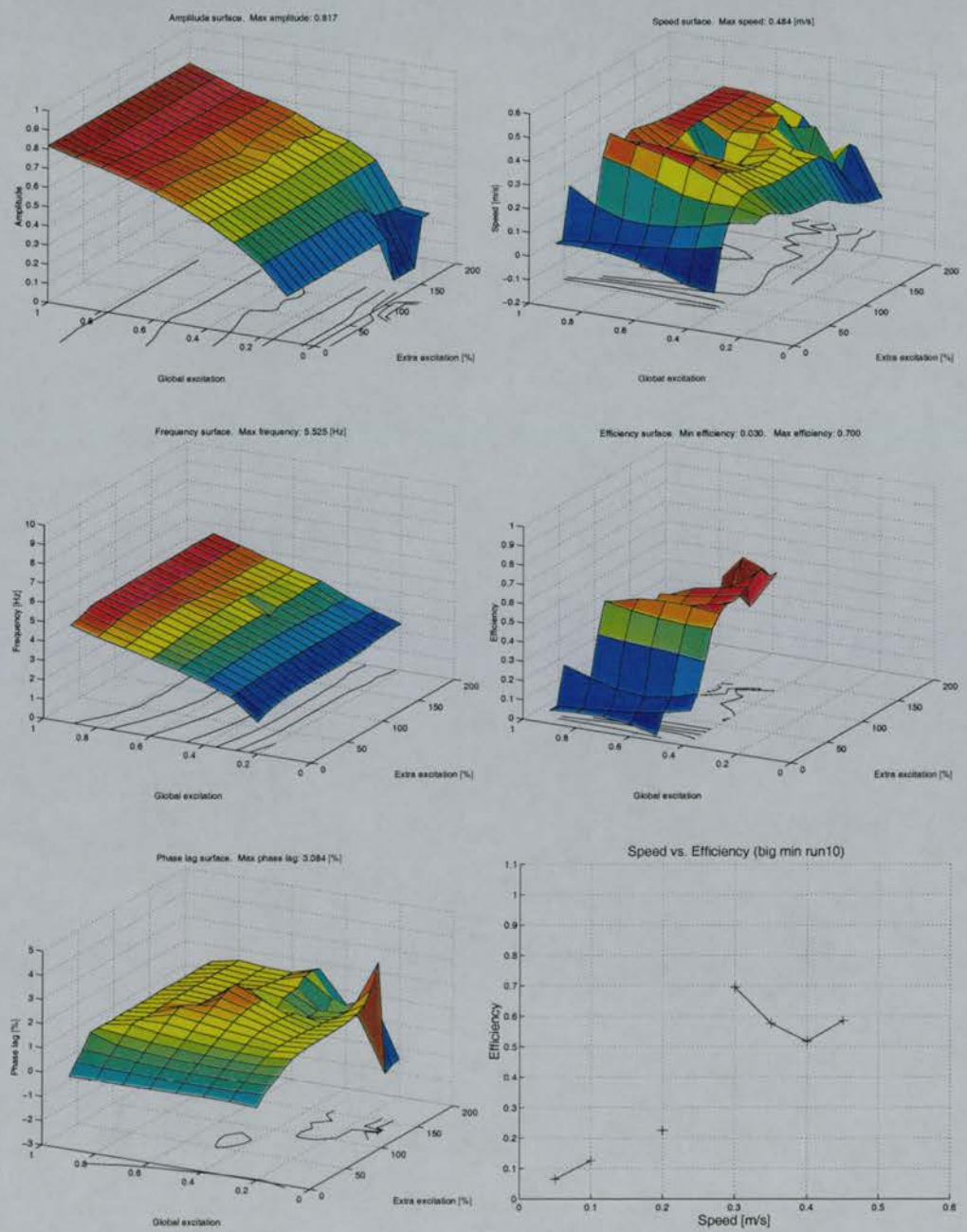


Figure C.4: Neural configuration and performance surface of run10 controller.

	MNl	EINl	LINl	CINl	CINr	LINr	EINr	MNr	BS
MNl	-	-0.4 [8, 3]	-	-3.2 [4, 1]	-	-	-	-	3.8
EINl	-	-0.8 [3, 10]	-	-3.8 [10, 4]	-0.7 [12, 12]	-	-0.9 [9, 10]	-	0.8
LINl	-	-	-	-	-	-	-	-	0.0
CINl	-	-	-	-	-3.7 [10, 10]	-	-3.5 [0, 5]	-	13.6
CINr	-	-3.5 [0, 5]	-	-3.7 [10, 10]	-	-	-	-	13.6
LINr	-	-	-	-	-	-	-	-	0.0
EINr	-	-0.9 [9, 10]	-	-0.7 [12, 12]	-3.8 [10, 4]	-	-0.8 [3, 10]	-	0.8
MNr	-	-	-	-	-3.2 [4, 1]	-	-0.4 [8, 3]	-	3.8

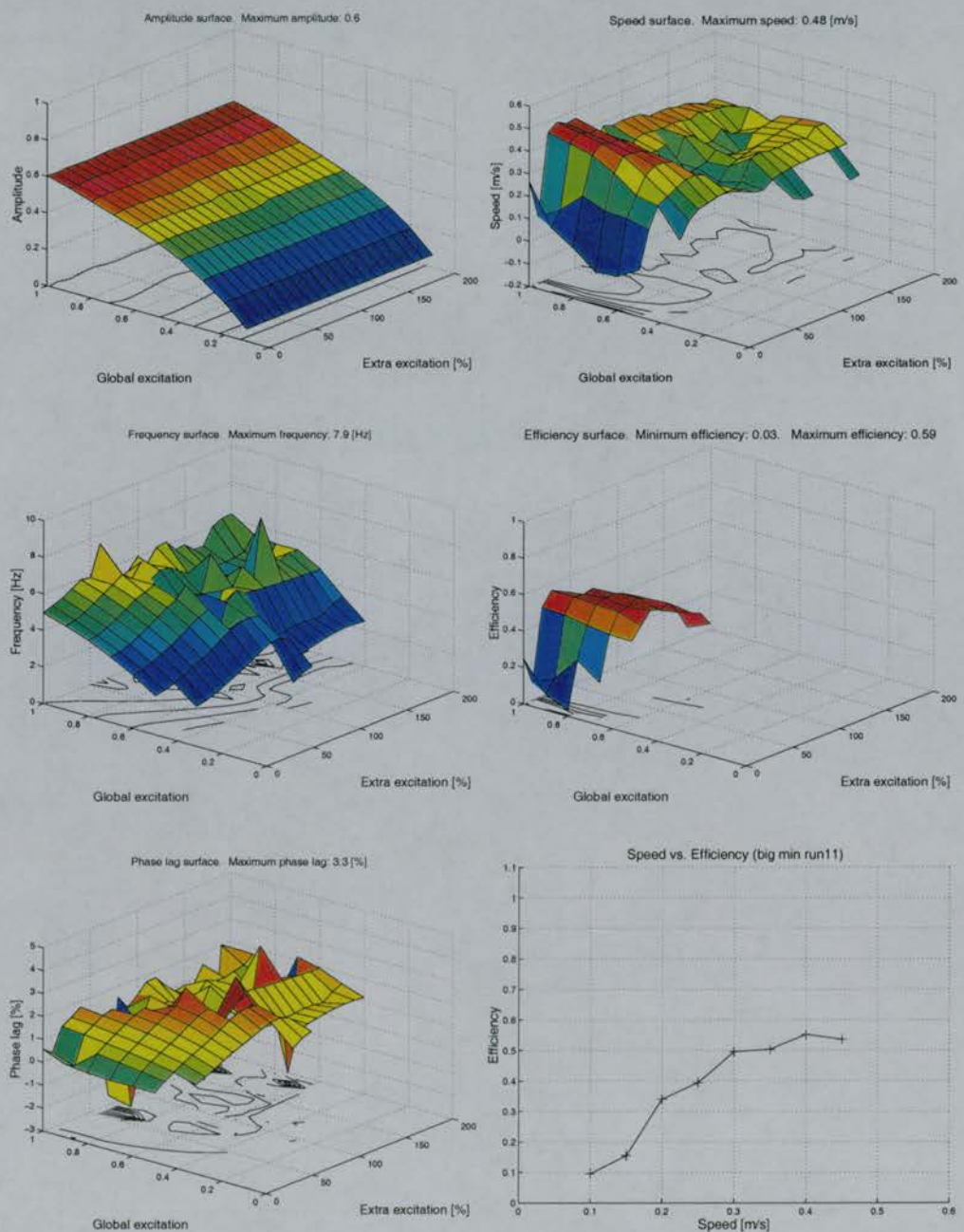


Figure C.5: Neural configuration and performance surface of run11 controller.

	MNl	EINl	LINl	CINl	CINr	LINr	EINr	MNr	BS
MNl	-	-0.4 [6, 3]	-	-3.2 [6, 1]	-	-	-	-	3.8
EINl	-	-0.8 [4, 5]	-	-3.8 [4, 5]	-0.7 [9, 7]	-	-0.9 [10, 5]	-	0.8
LINl	-	-	-	-	-	-	-	-	0.0
CINl	-	-	-	-	-3.7 [5, 10]	-	-3.5 [5, 3]	-	13.6
CINr	-	-3.5 [5, 3]	-	-3.7 [5, 10]	-	-	-	-	13.6
LINr	-	-	-	-	-	-	-	-	0.0
EINr	-	-0.9 [10, 5]	-	-0.7 [9, 7]	-3.8 [4, 5]	-	-0.8 [4, 5]	-	0.8
MNr	-	-	-	-	-3.2 [6, 1]	-	-0.4 [6, 3]	-	3.8

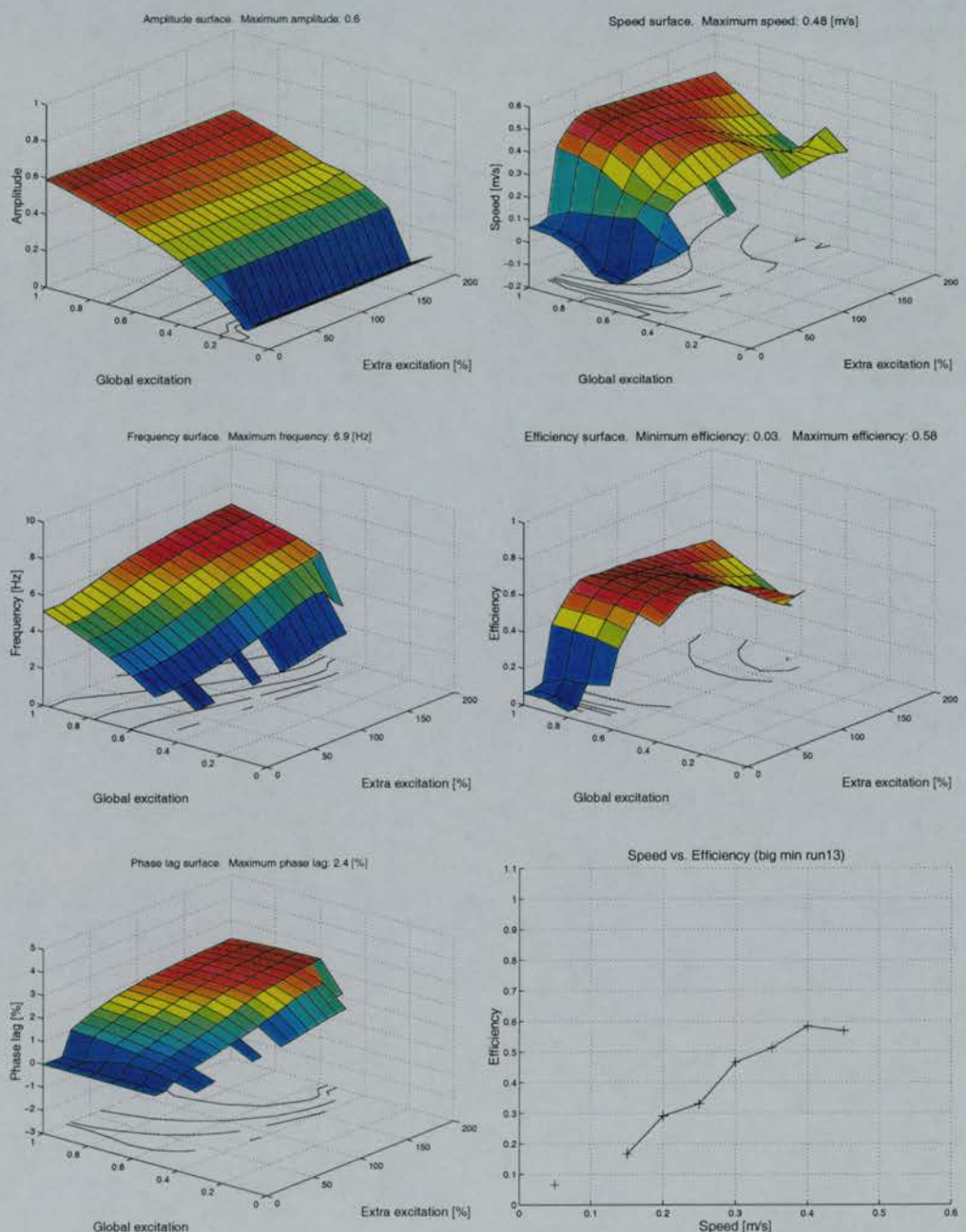


Figure C.6: Neural configuration and performance surface of run13 controller.

	MNl	EINl	LINl	CINl	CINr	LINr	EINr	MNr	BS
MNl	-	-	-	-	-2.5 [11, 4]	-	-	-	5.5
EINl	-	-	-4.2 [9, 9]	-	-2.5 [10, 7]	-	-	-	0.9
LINl	-	14.6 [9, 6]	-	-	-	-0.6 [7, 5]	-	-	4.8
CINl	-	-	-4.8 [4, 2]	-	-1.3 [2, 4]	-	-	-	7.8
CINr	-	-	-	-1.3 [2, 4]	-	-4.8 [4, 2]	-	-	7.8
LINr	-	-	-0.6 [7, 5]	-	-	-	14.6 [9, 6]	-	4.8
EINr	-	-	-	-2.5 [10, 7]	-	-4.2 [9, 9]	-	-	0.9
MNr	-	-	-	-2.5 [11, 4]	-	-	-	-	5.5

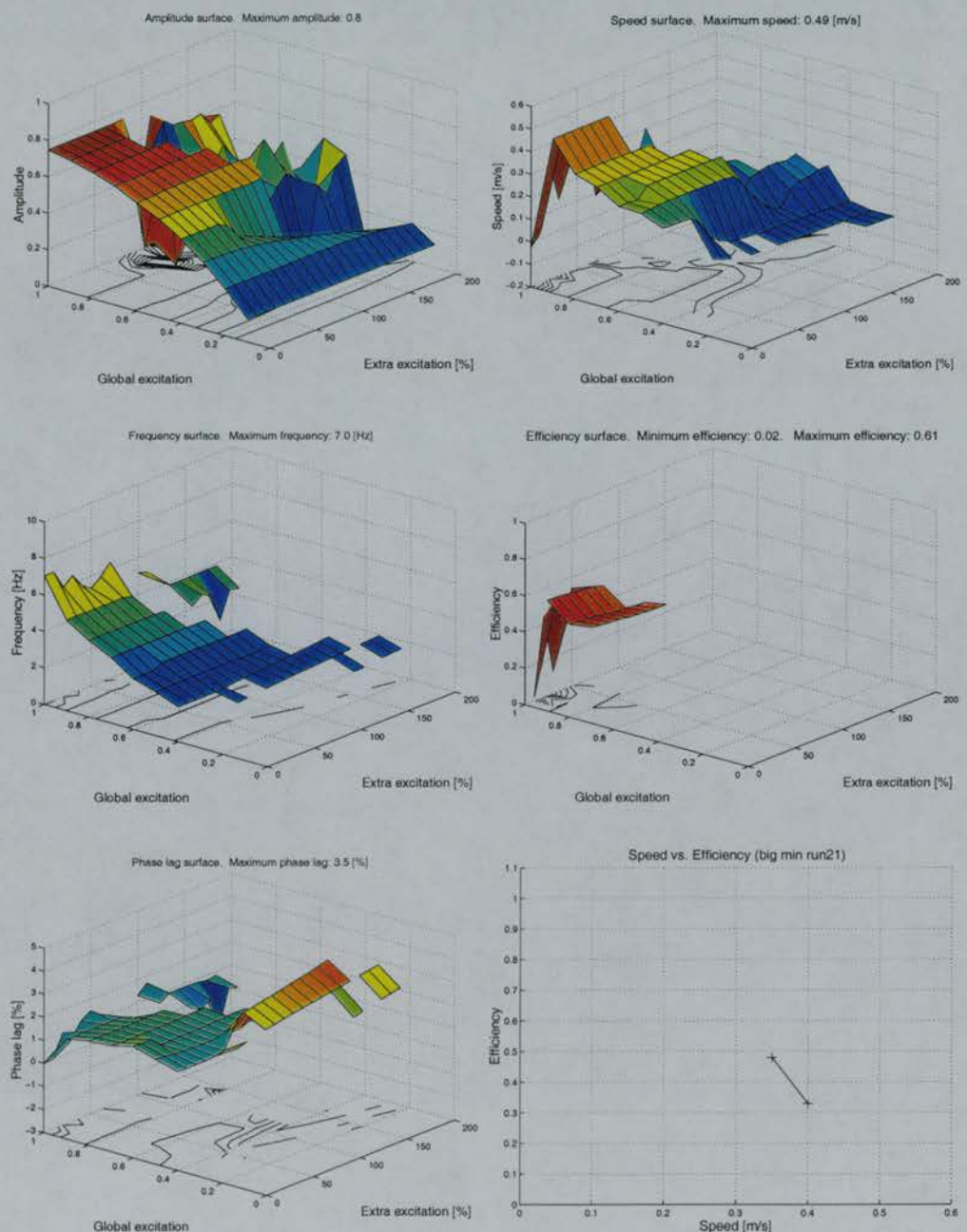


Figure C.7: Neural configuration and performance surface of run21 controller.

	MNl	EINl	LINl	CINl	CINr	LINr	EINr	MNr	BS
MNl	-	-	-	-	-2.5 [10, 4]	-	-	-	5.5
EINl	-	-	-4.2 [8, 12]	-	-2.5 [10, 8]	-	-	-	0.9
LINl	-	14.6 [11, 7]	-	-	-	-0.6 [9, 5]	-	-	4.8
CINl	-	-	-4.8 [4, 2]	-	-1.3 [2, 5]	-	-	-	7.8
CINr	-	-	-	-1.3 [2, 5]	-	-4.8 [4, 2]	-	-	7.8
LINr	-	-	-0.6 [9, 5]	-	-	-	14.6 [11, 7]	-	4.8
EINr	-	-	-	-2.5 [10, 8]	-	-4.2 [8, 12]	-	-	0.9
MNr	-	-	-	-2.5 [10, 4]	-	-	-	-	5.5

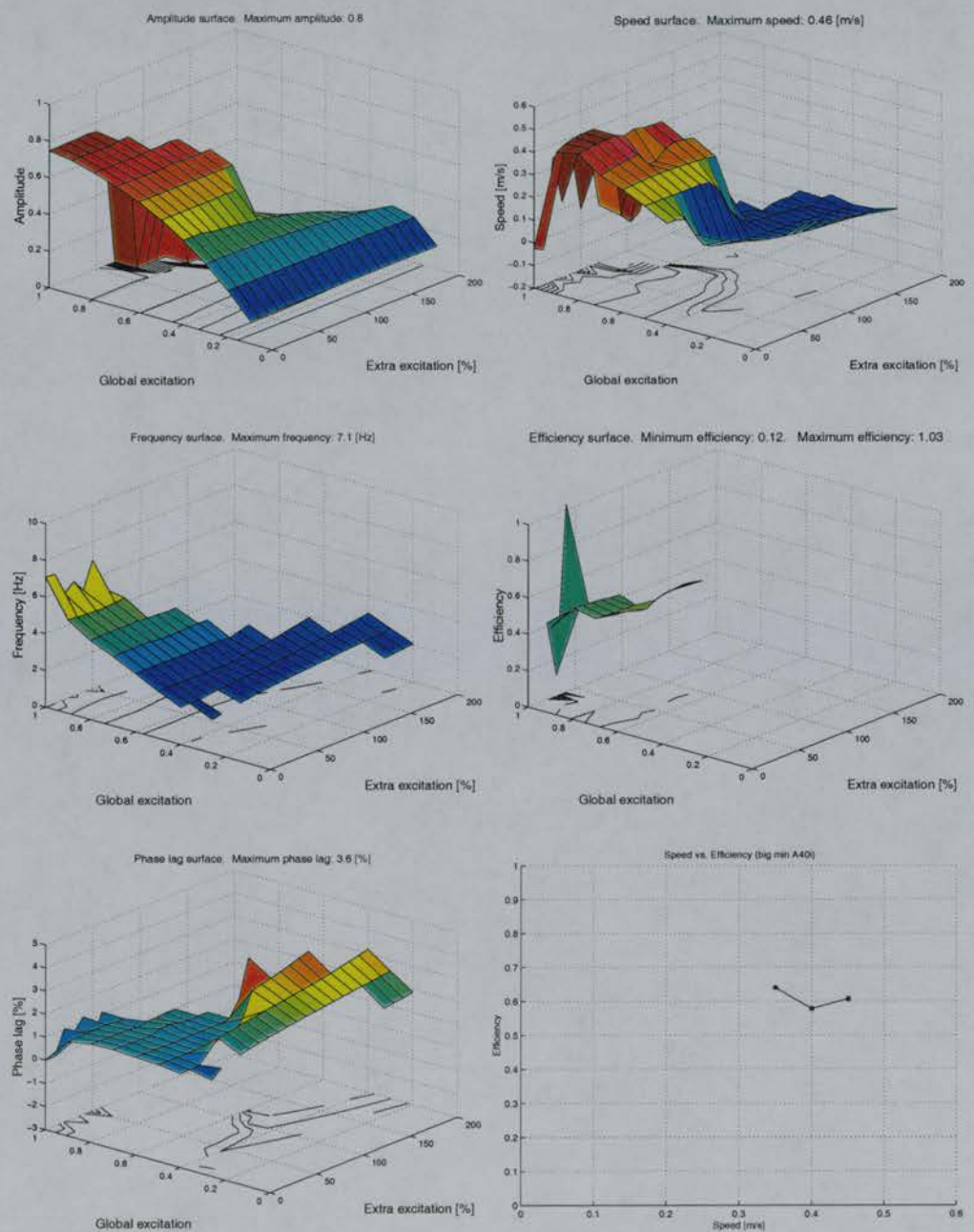


Figure C.8: Neural configuration and performance surface of run40 controller.

	MNl	EINl	LINl	CINl	CINr	LINr	EINr	MNr	BS
MNl	-	-	-	-	-2.5 [12, 9]	-	-	-	5.5
EINl	-	-	-4.2 [11, 10]	-	-2.5 [9, 3]	-	-	-	0.9
LINl	-	14.6 [7, 11]	-	-	-	-0.6 [10, 10]	-	-	4.8
CINl	-	-	-4.8 [8, 6]	-	-1.3 [4, 7]	-	-	-	7.8
CINr	-	-	-	-1.3 [4, 7]	-	-4.8 [8, 6]	-	-	7.8
LINr	-	-	-0.6 [10, 10]	-	-	-	14.6 [7, 11]	-	4.8
EINr	-	-	-	-2.5 [9, 3]	-	-4.2 [11, 10]	-	-	0.9
MNr	-	-	-	-2.5 [12, 9]	-	-	-	-	5.5

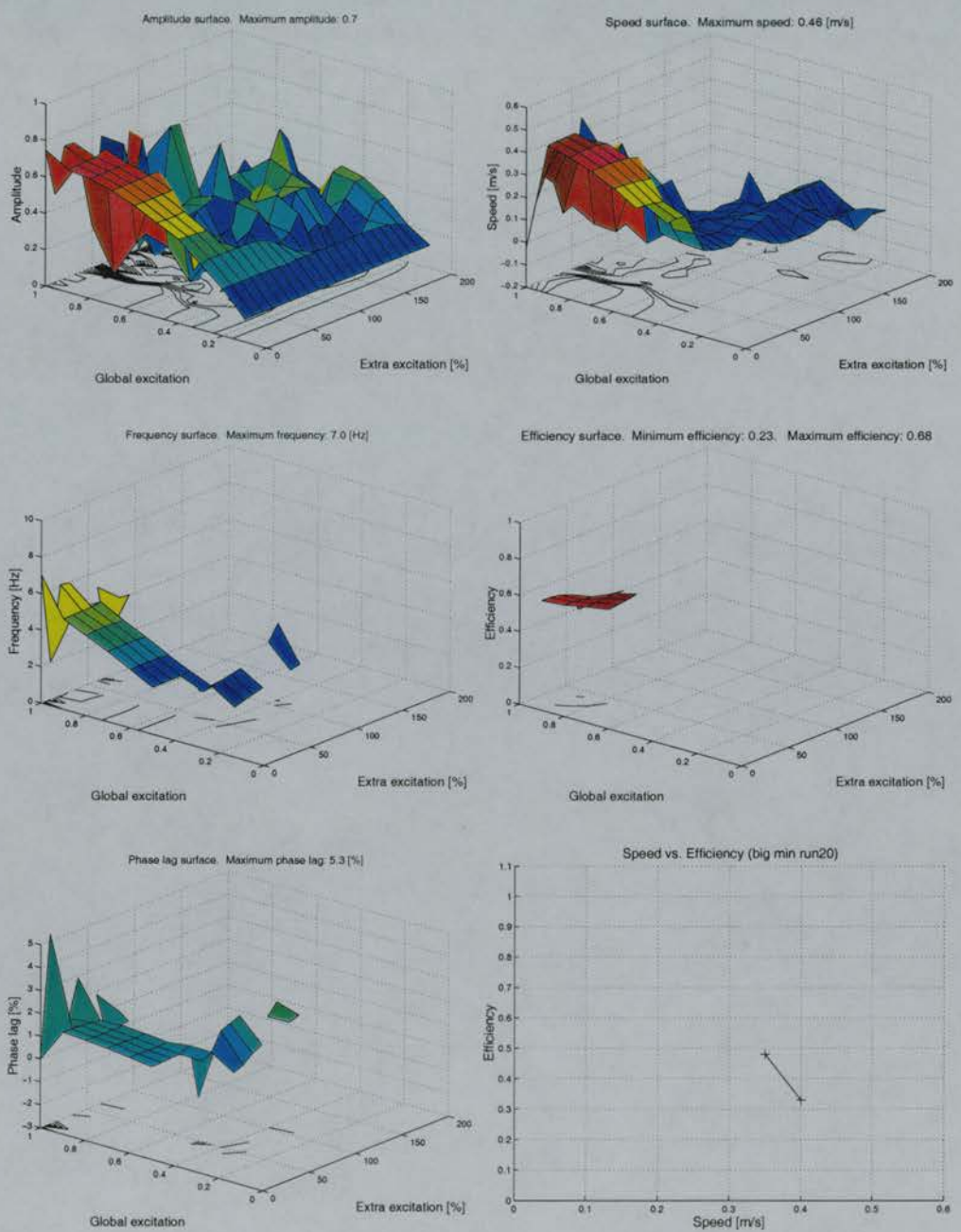


Figure C.9: Neural configuration and performance surface of run20 controller.

	MNl	EINl	LINl	CINl	CINr	LINr	EINr	MNr	BS
MNl	-	-	-	-	-2.5 [3, 9]	-	-	-	5.5
EINl	-	-	-4.2 [1, 6]	-	-2.5 [9, 11]	-	-	-	0.9
LINl	-	14.6 [12, 8]	-	-	-	-0.6 [7, 10]	-	-	4.8
CINl	-	-	-4.8 [0, 3]	-	-1.3 [0, 6]	-	-	-	7.8
CINr	-	-	-	-1.3 [0, 6]	-	-4.8 [0, 3]	-	-	7.8
LINr	-	-	-0.6 [7, 10]	-	-	-	14.6 [12, 8]	-	4.8
EINr	-	-	-	-2.5 [9, 11]	-	-4.2 [1, 6]	-	-	0.9
MNr	-	-	-	-2.5 [3, 9]	-	-	-	-	5.5

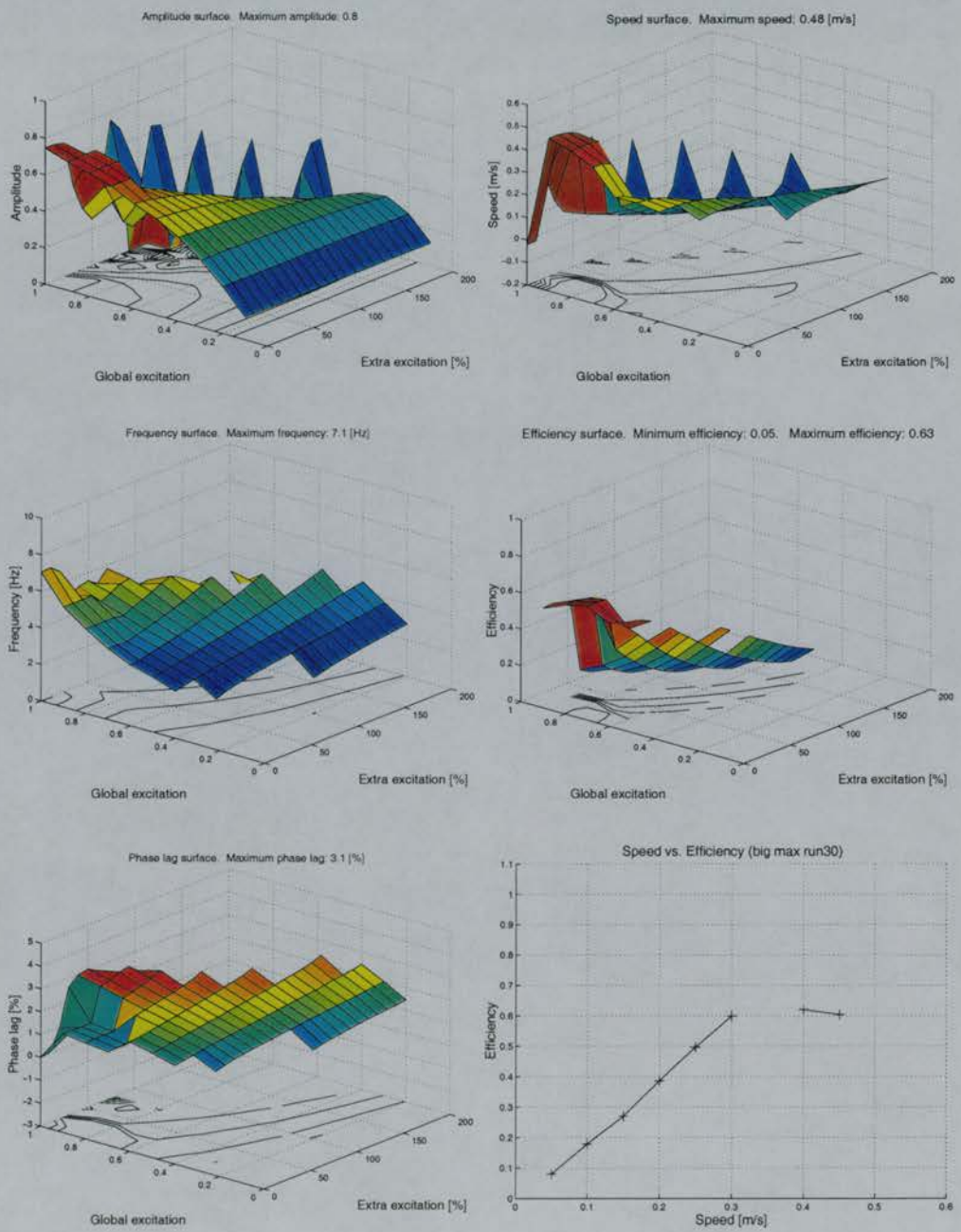


Figure C.10: Neural configuration and performance surface of run30 controller.

Appendix D

Speed vs. efficiency curve at various body scales (bigmax controllers)

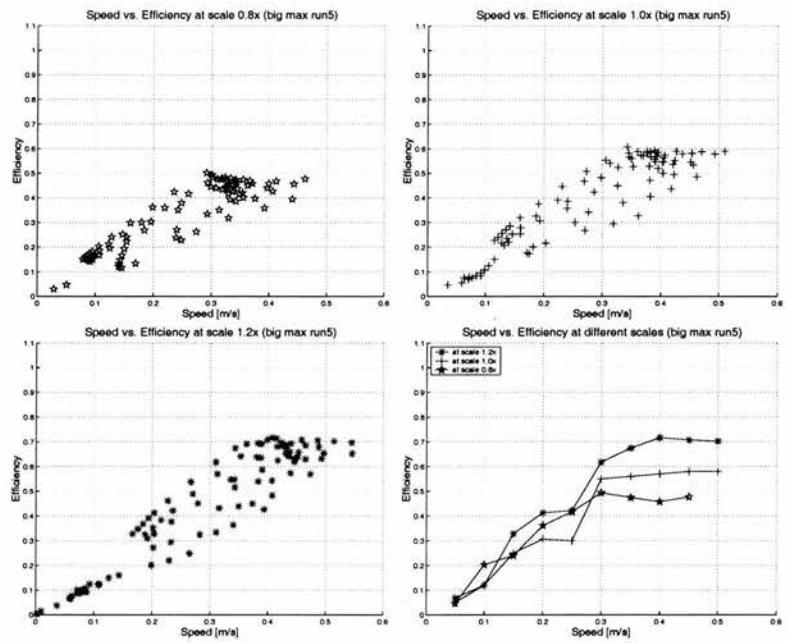


Figure D.1: Speed vs. Efficiency curves at various body scales for the big max run5 controller.

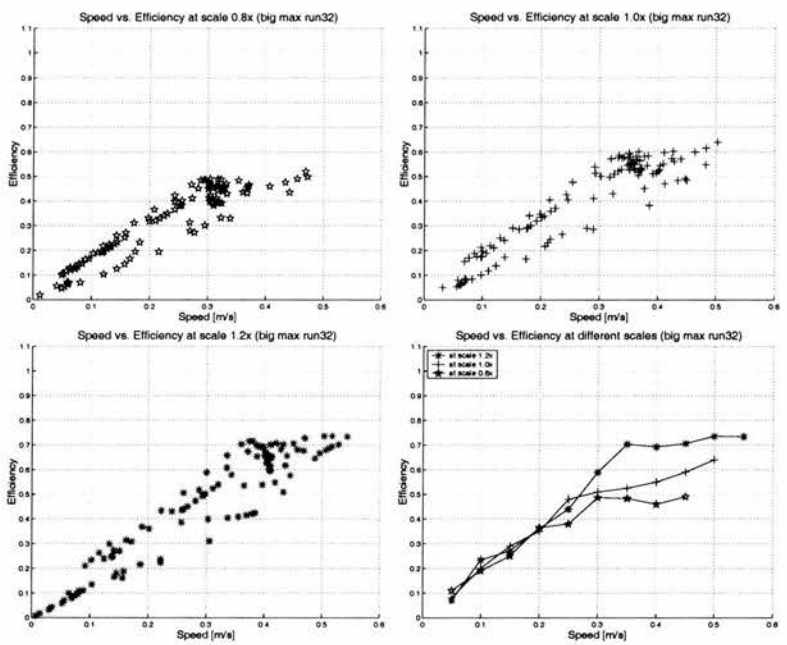


Figure D.2: Speed vs. Efficiency curves at various body scales for the big max run32 controller.

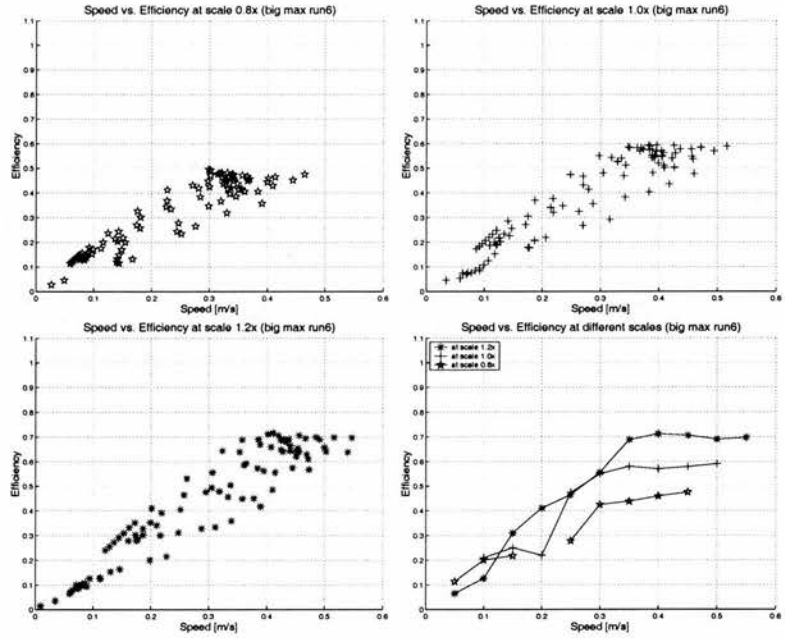


Figure D.3: Speed vs. Efficiency curves at various body scales for the big max run6 controller.

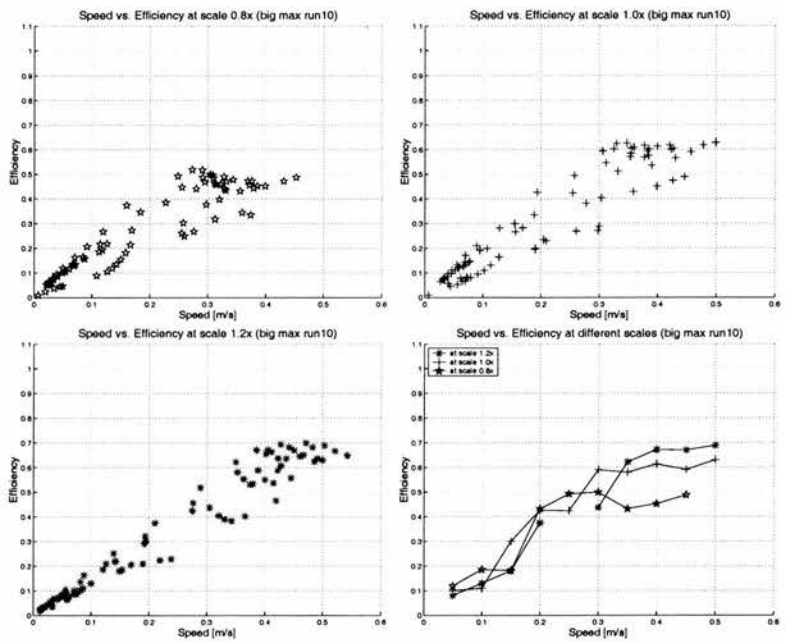


Figure D.4: Speed vs. Efficiency curves at various body scales for the big max run10 controller.

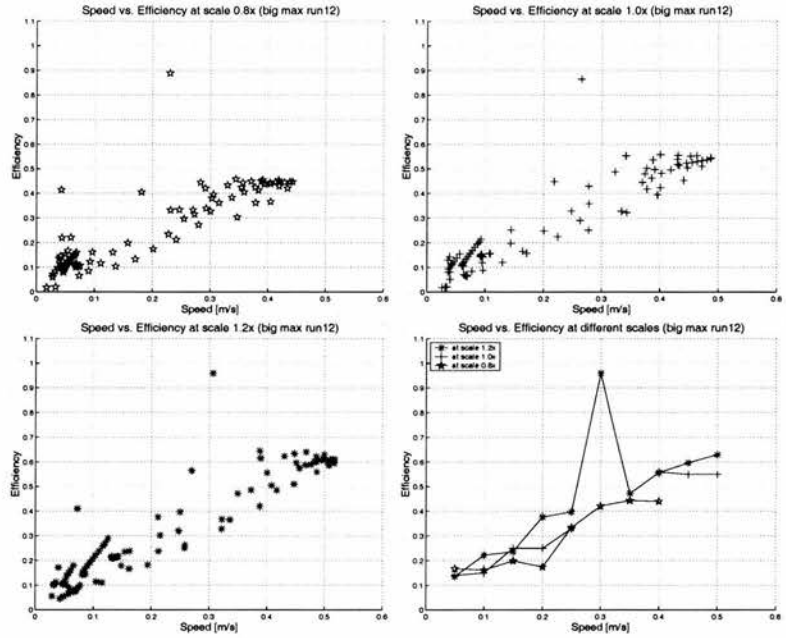


Figure D.5: Speed vs. Efficiency curves at various body scales for the big max run12 controller.

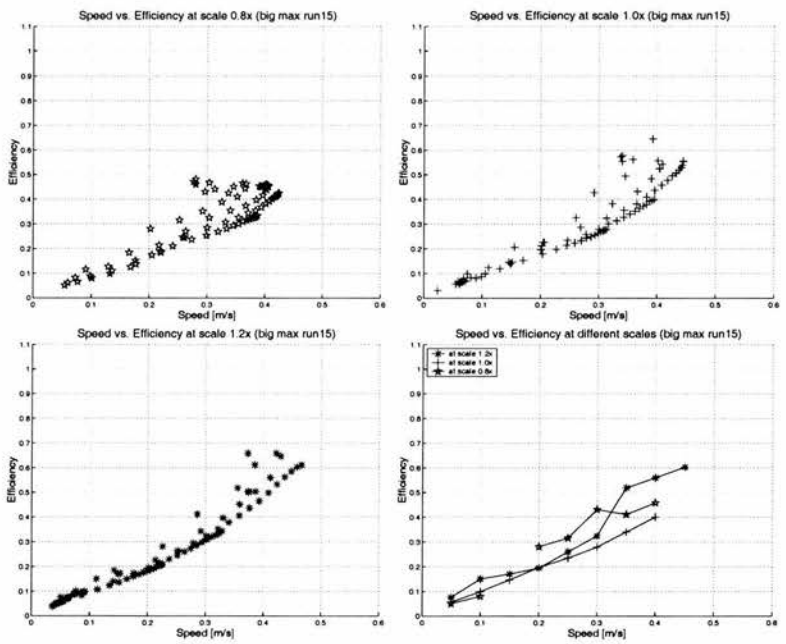


Figure D.6: Speed vs. Efficiency curves at various body scales for the big max run15 controller.

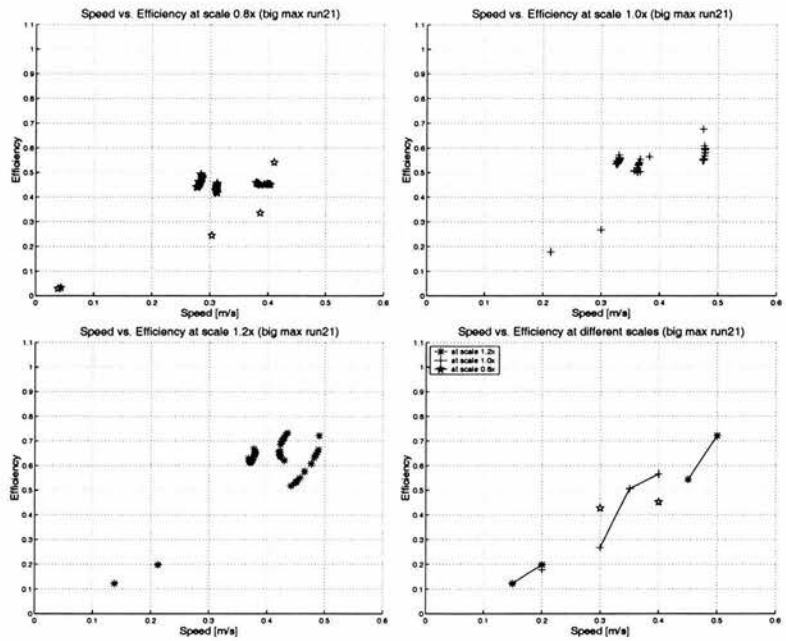


Figure D.7: Speed vs. Efficiency curves at various body scales for the big max run21 controller.

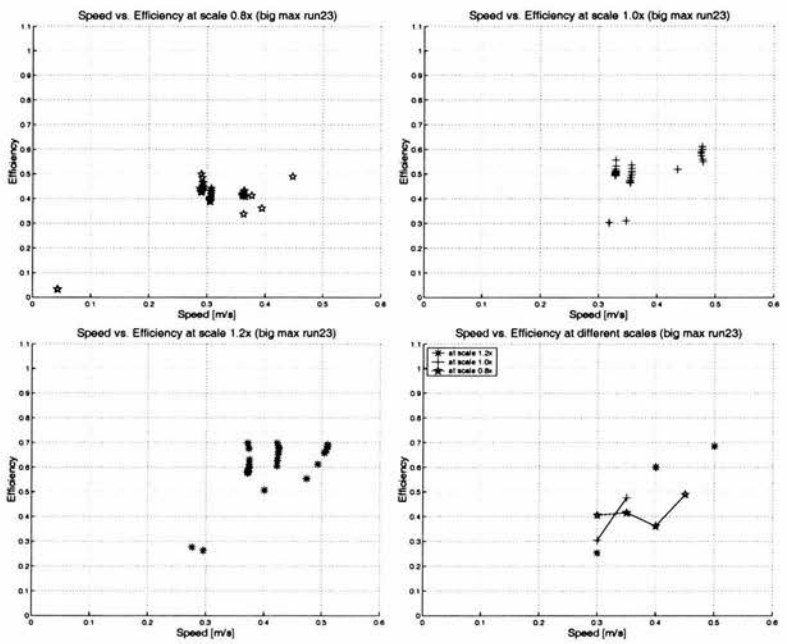


Figure D.8: Speed vs. Efficiency curves at various body scales for the big max run23 controller.

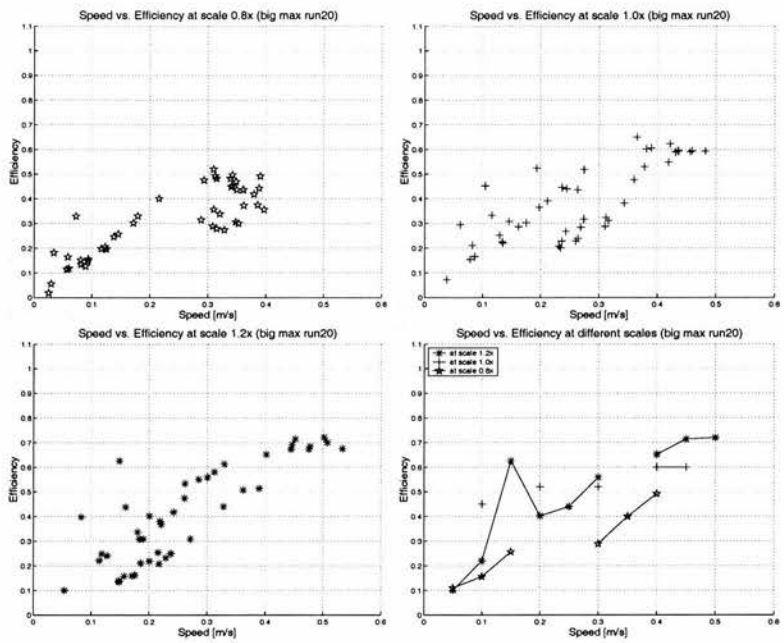


Figure D.9: Speed vs. Efficiency curves at various body scales for the big max run20 controller.

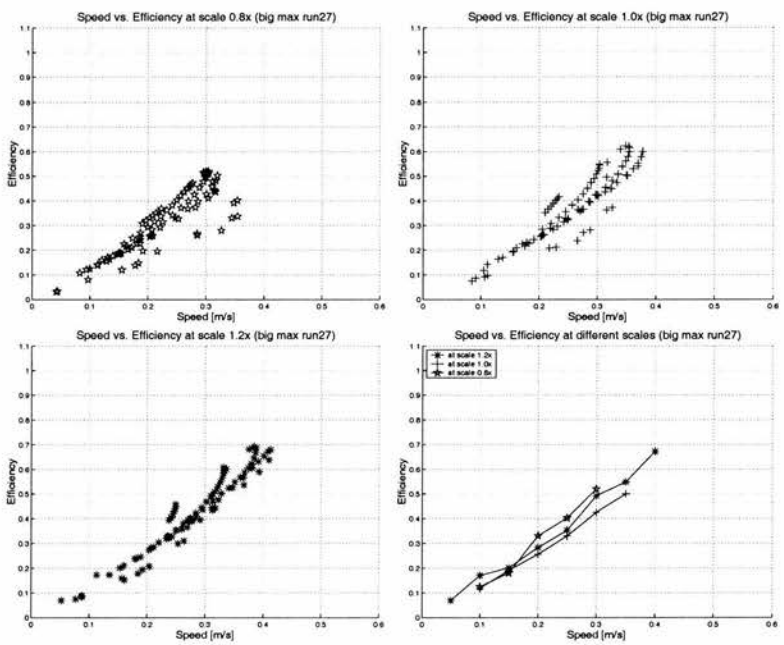


Figure D.10: Speed vs. Efficiency curves at various body scales for the big max run27 controller.

Appendix E

Speed vs. efficiency curve at various body scales (bigmin controllers)

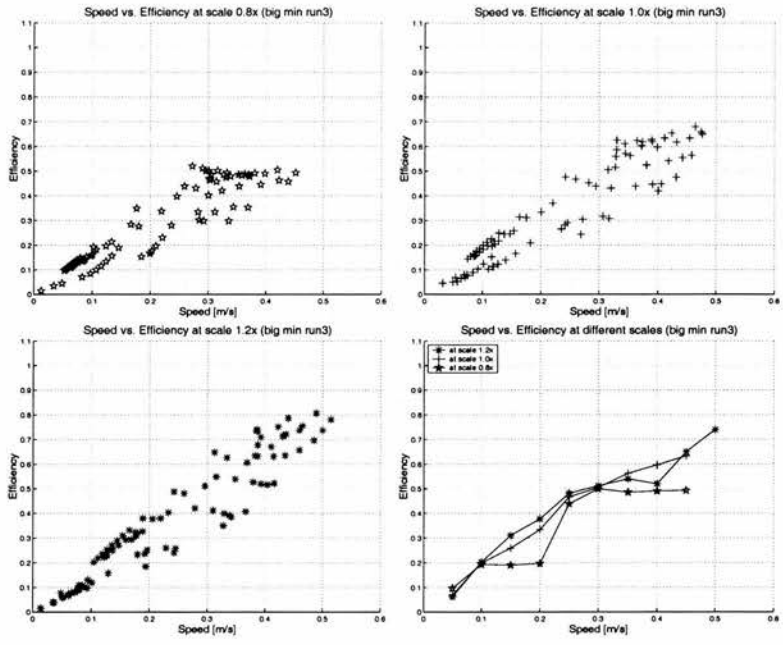


Figure E.1: Speed vs. Efficiency curves at various body scales for the big min run3 controller.

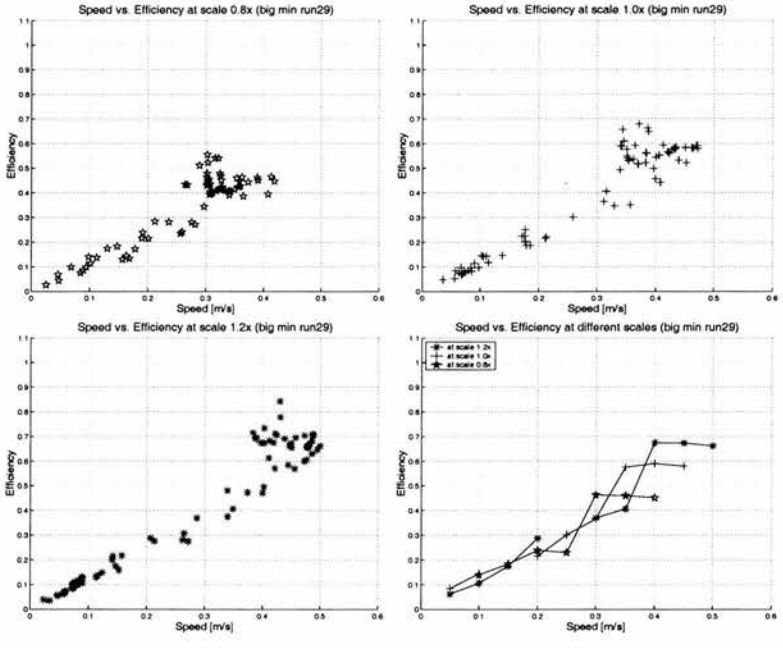


Figure E.2: Speed vs. Efficiency curves at various body scales for the big min run29 controller.

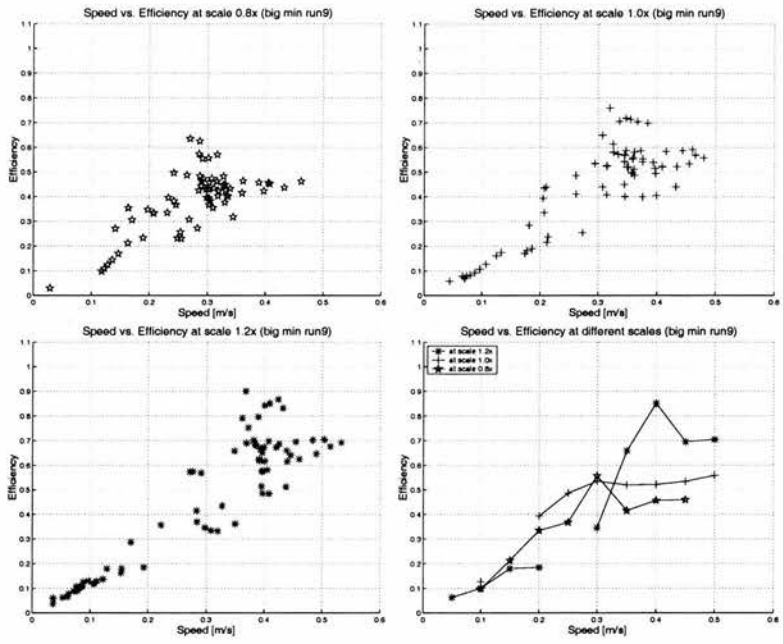


Figure E.3: Speed vs. Efficiency curves at various body scales for the big min run9 controller.

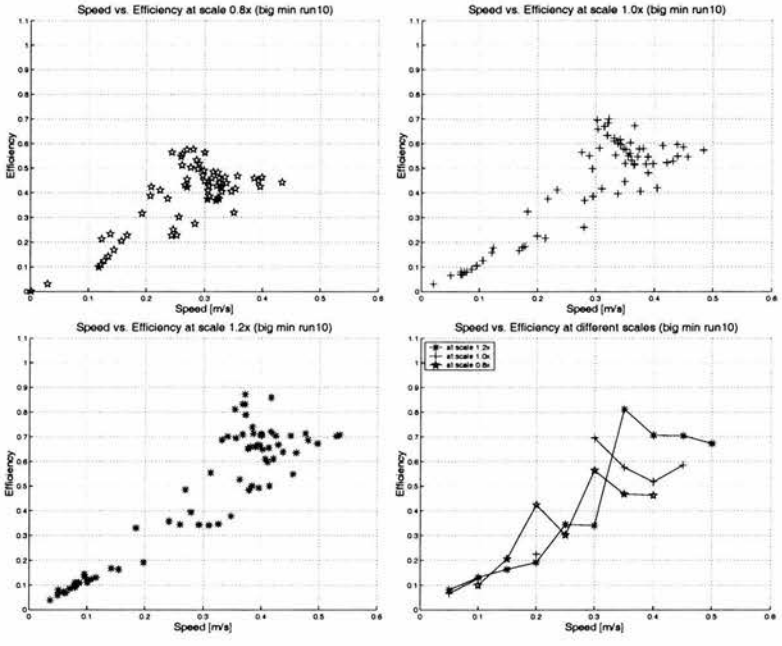


Figure E.4: Speed vs. Efficiency curves at various body scales for the big min run10 controller.

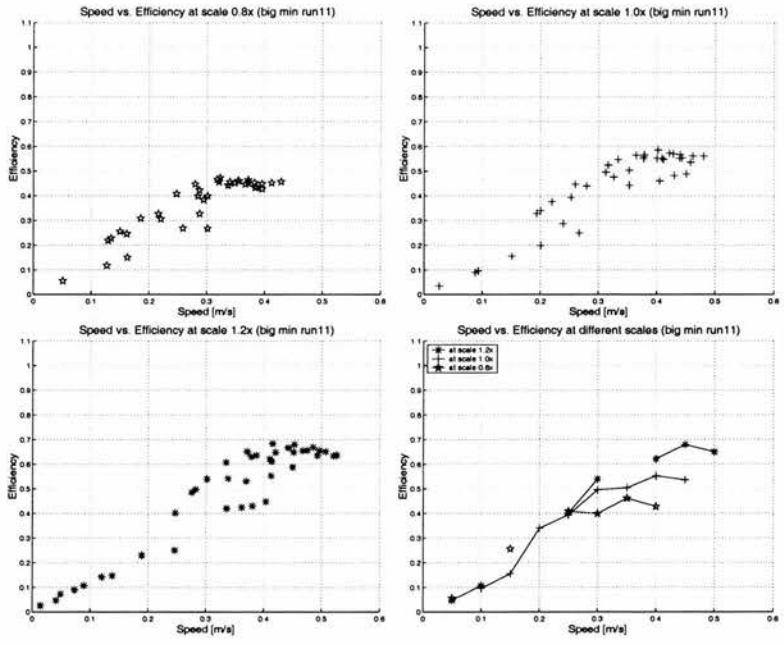


Figure E.5: Speed vs. Efficiency curves at various body scales for the big min run11 controller.

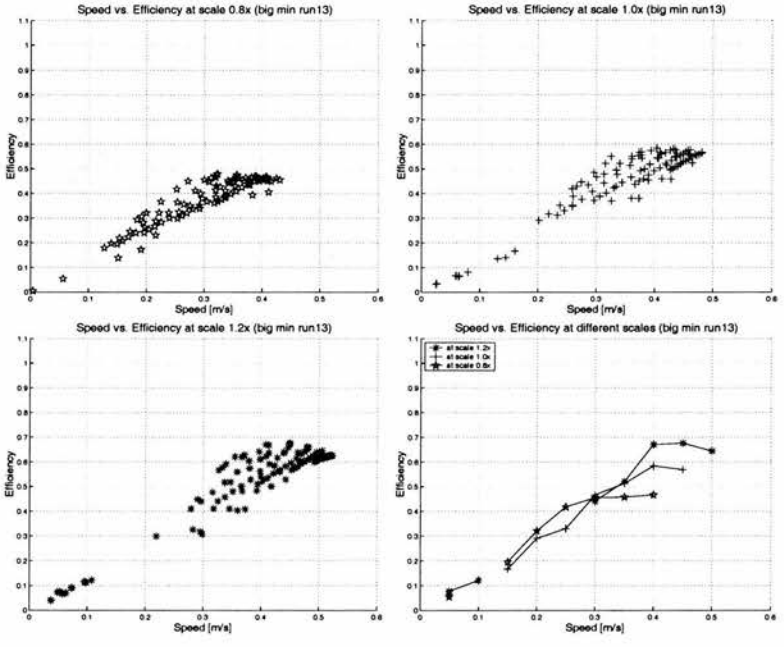


Figure E.6: Speed vs. Efficiency curves at various body scales for the big min run13 controller.

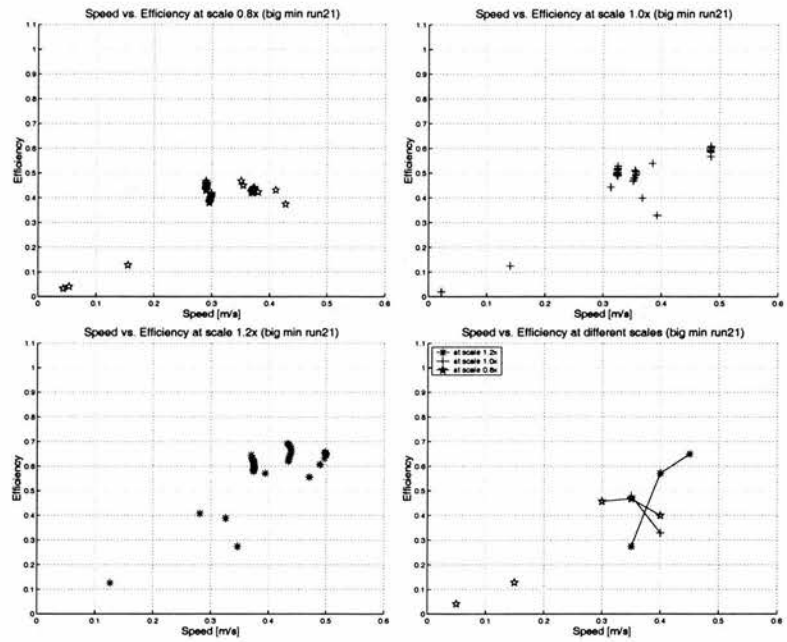


Figure E.7: Speed vs. Efficiency curves at various body scales for the big min run21 controller.

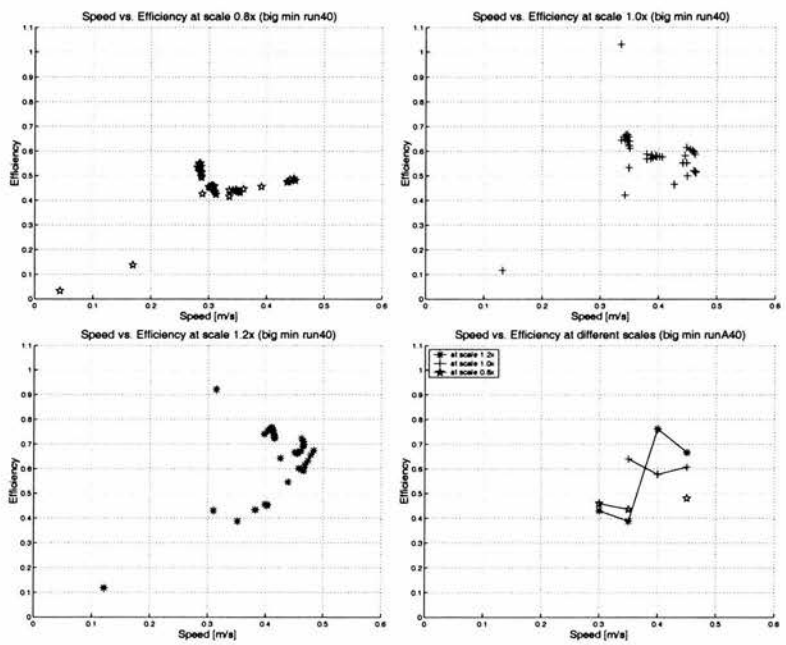


Figure E.8: Speed vs. Efficiency curves at various body scales for the big min run40 controller.

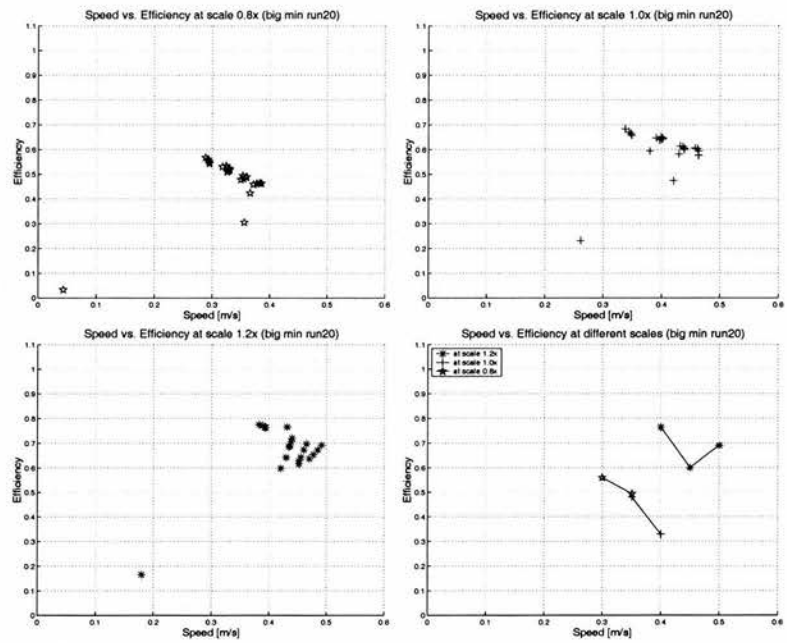


Figure E.9: Speed vs. Efficiency curves at various body scales for the big min run20 controller.

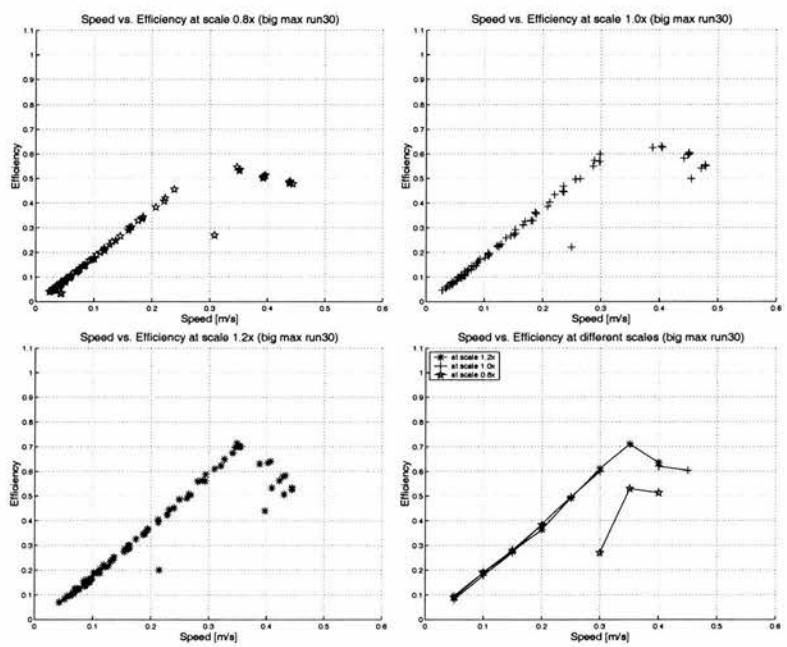


Figure E.10: Speed vs. Efficiency curves at various body scales for the big min run30 controller.

Appendix F

Robustness of the swimming CPGs against random variation in neural connections

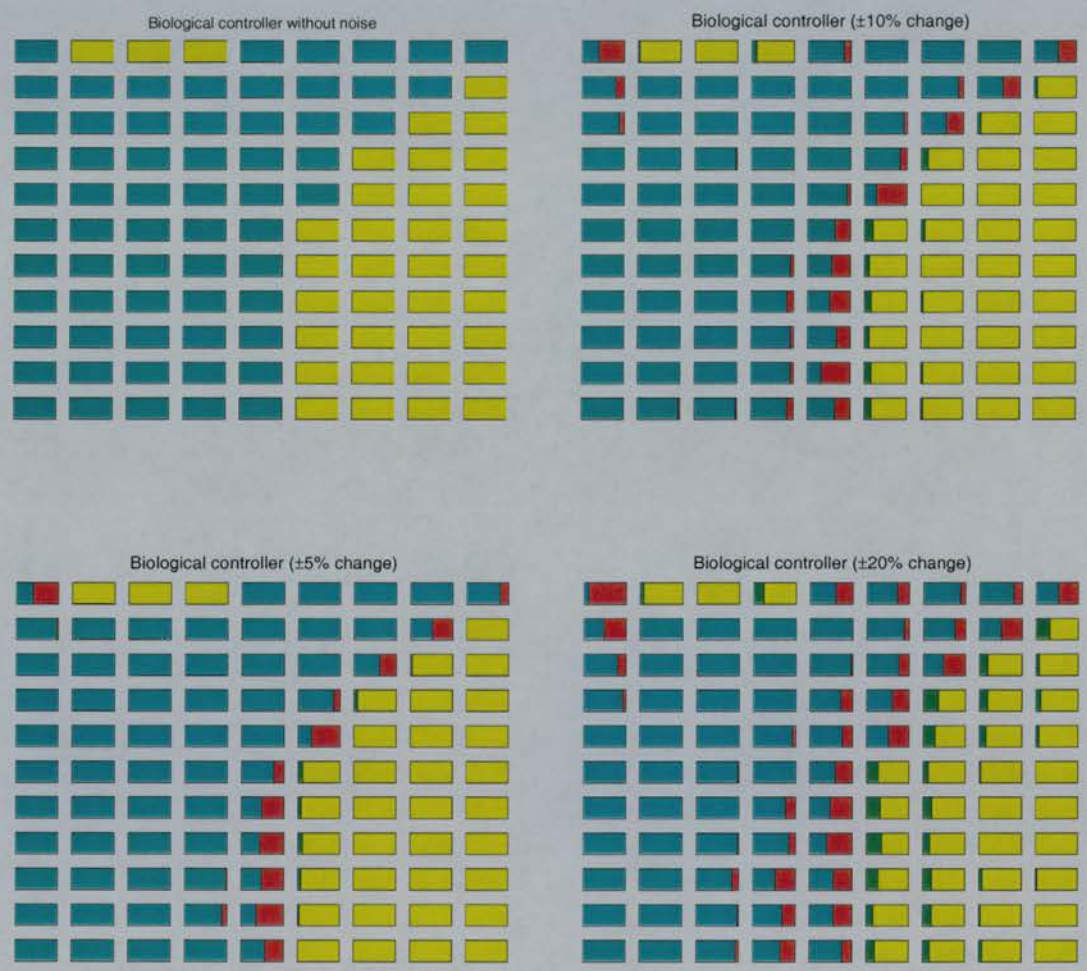


Figure F.1: Results of Analysis 1 (biological controller).

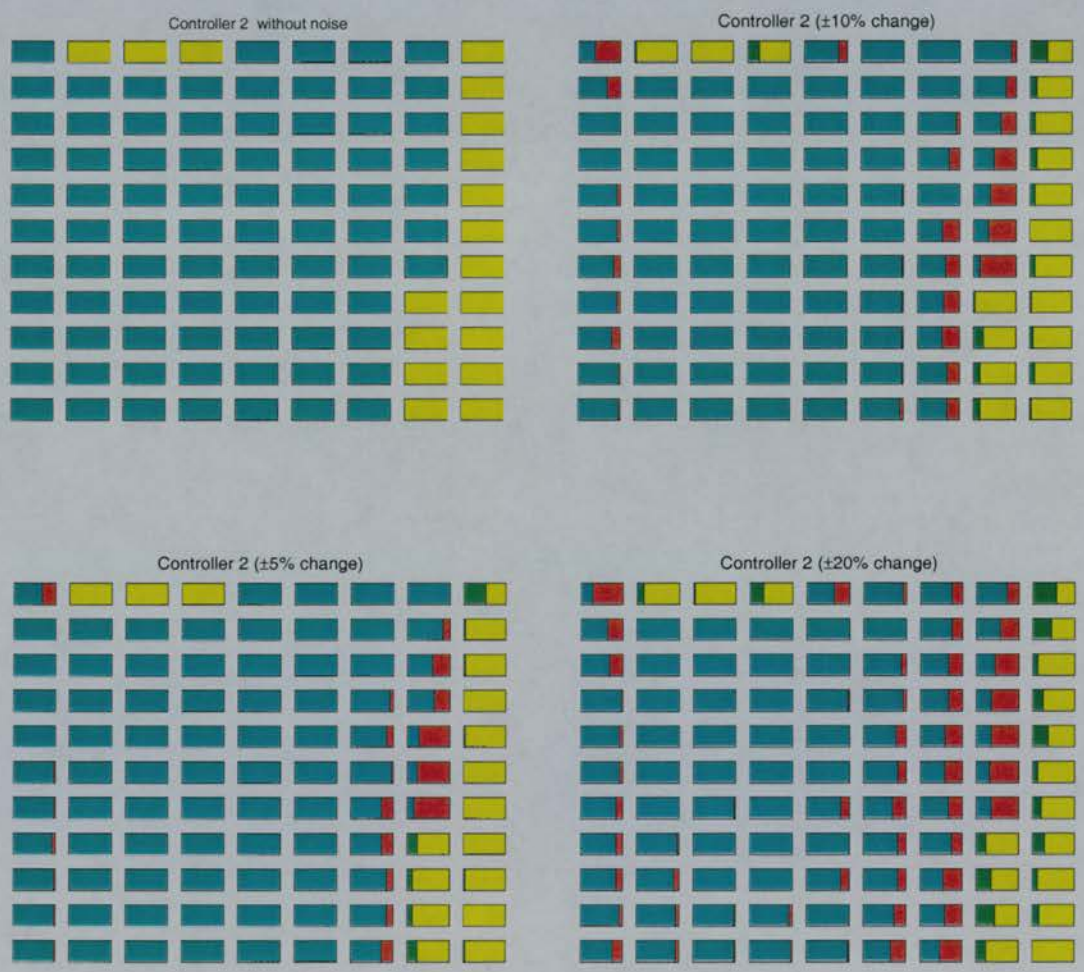


Figure F.2: Results of Analysis 1 (controller 2).

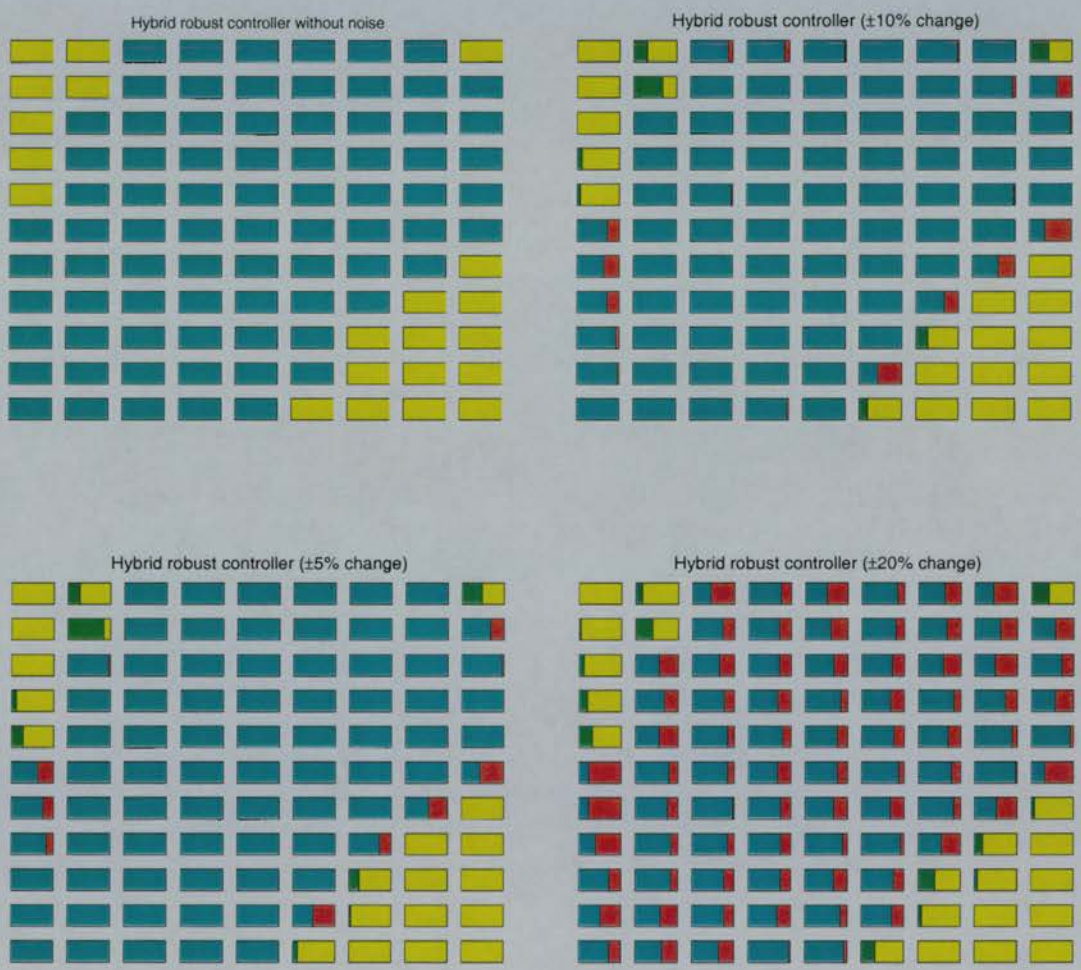


Figure F.3: Results of Analysis 1 (hybrid robust controller).

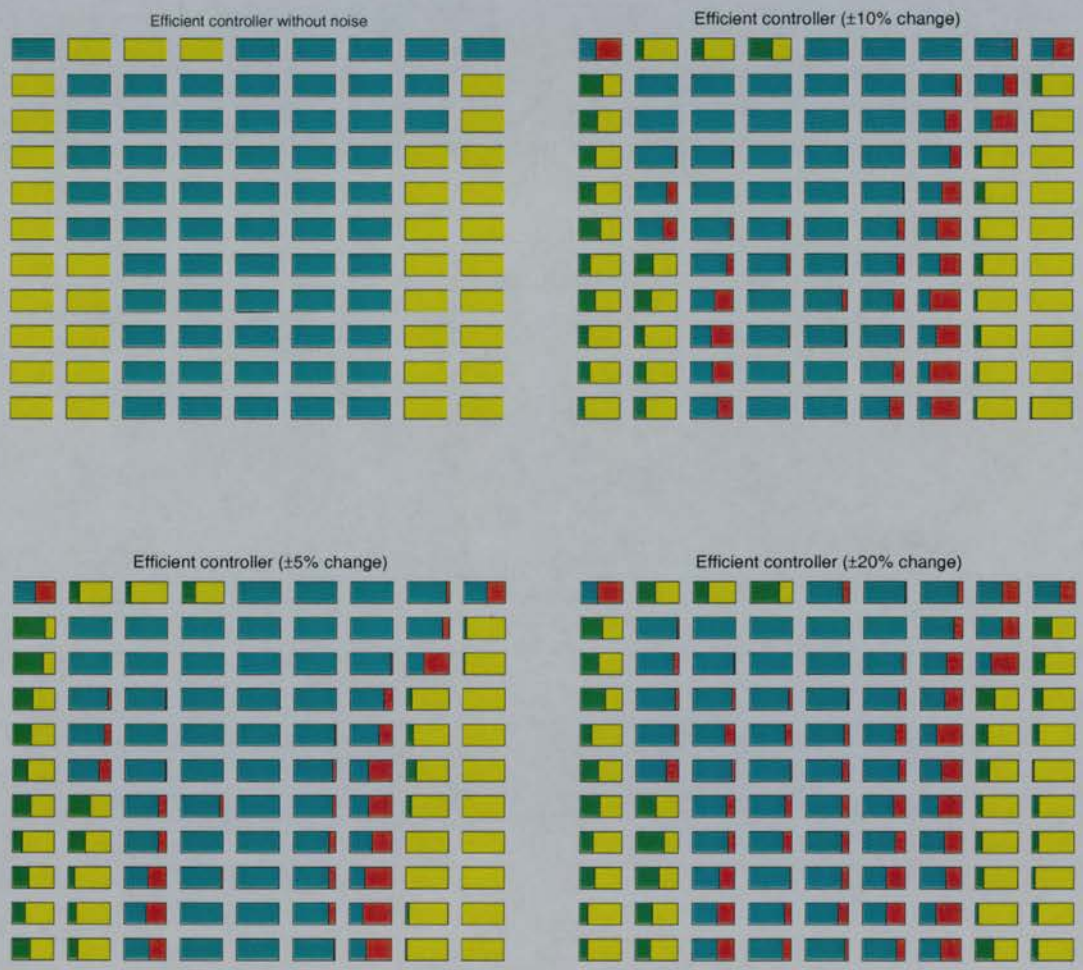


Figure F.4: Results of Analysis 1 (efficient controller).

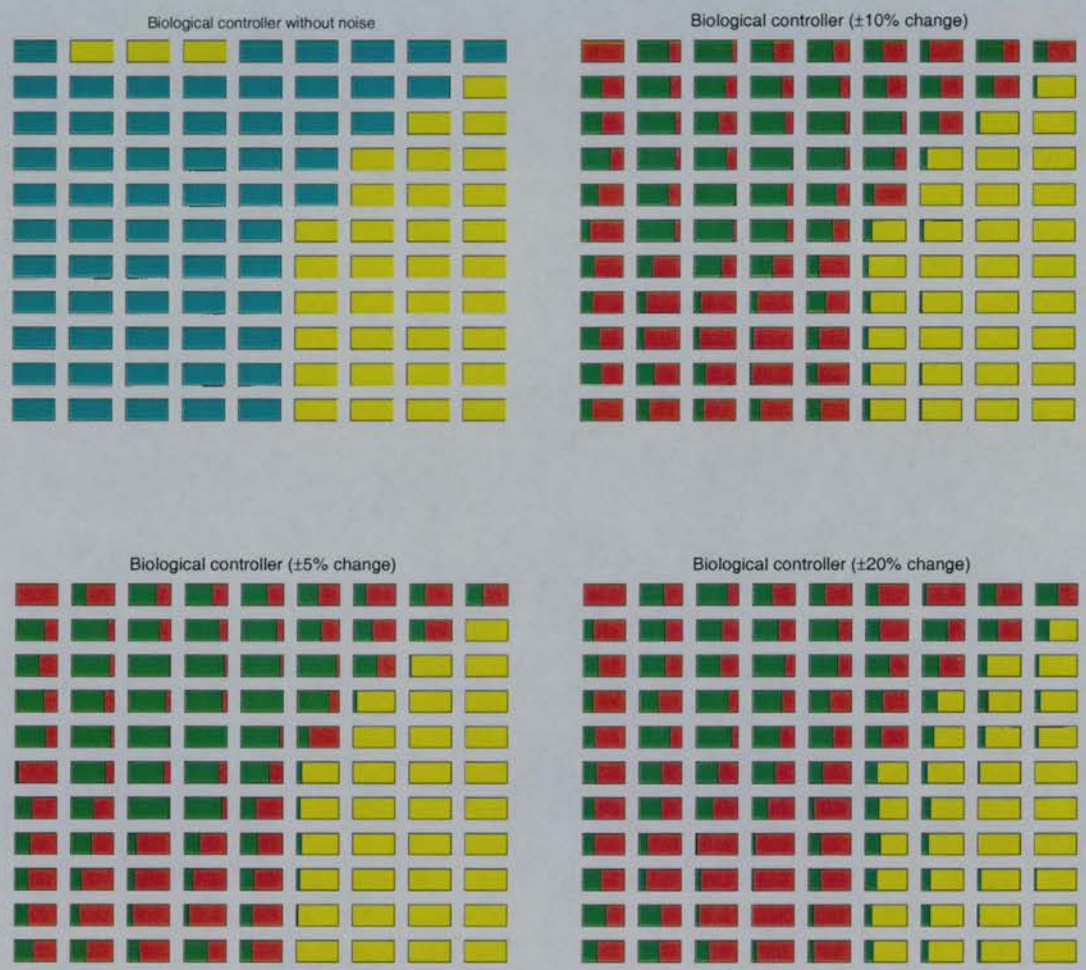


Figure F.5: Results of Analysis 2 (biological controller).

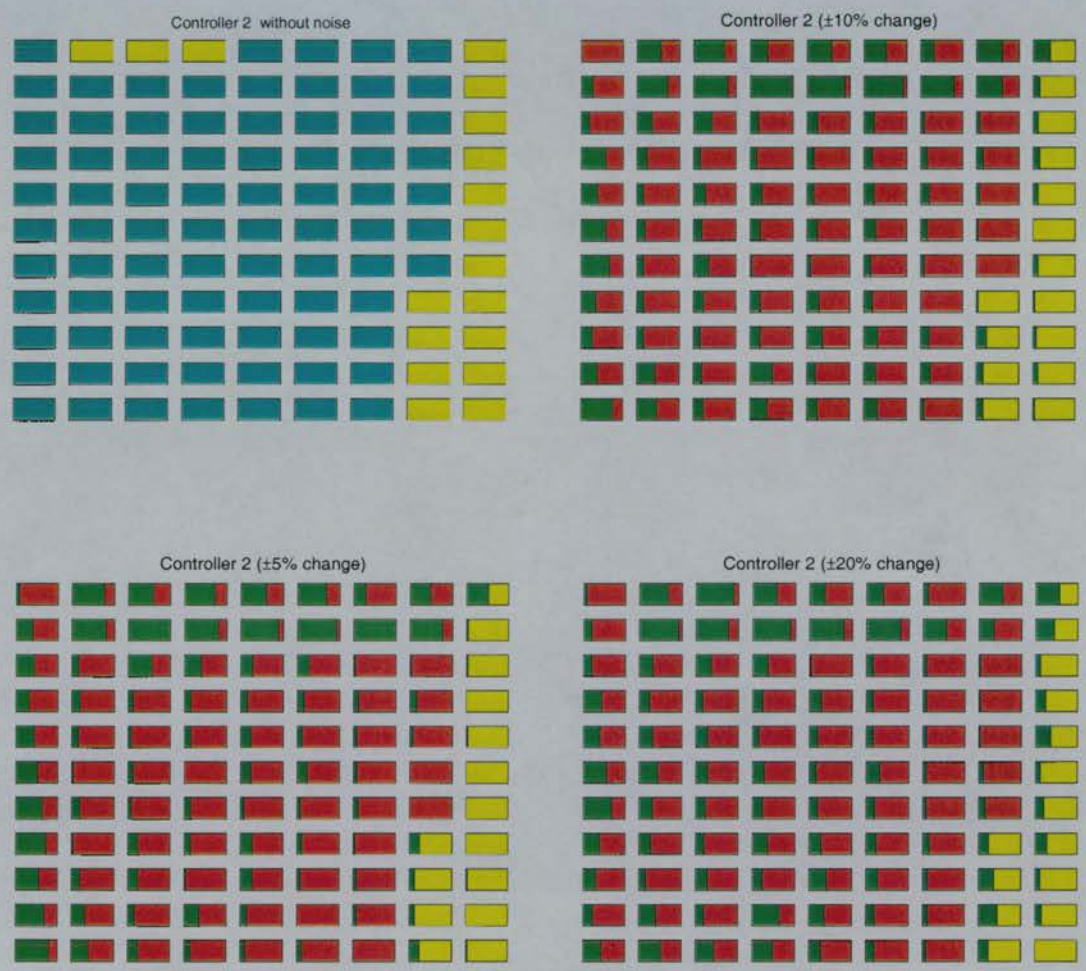


Figure F.6: Results of Analysis 2 (controller 2).

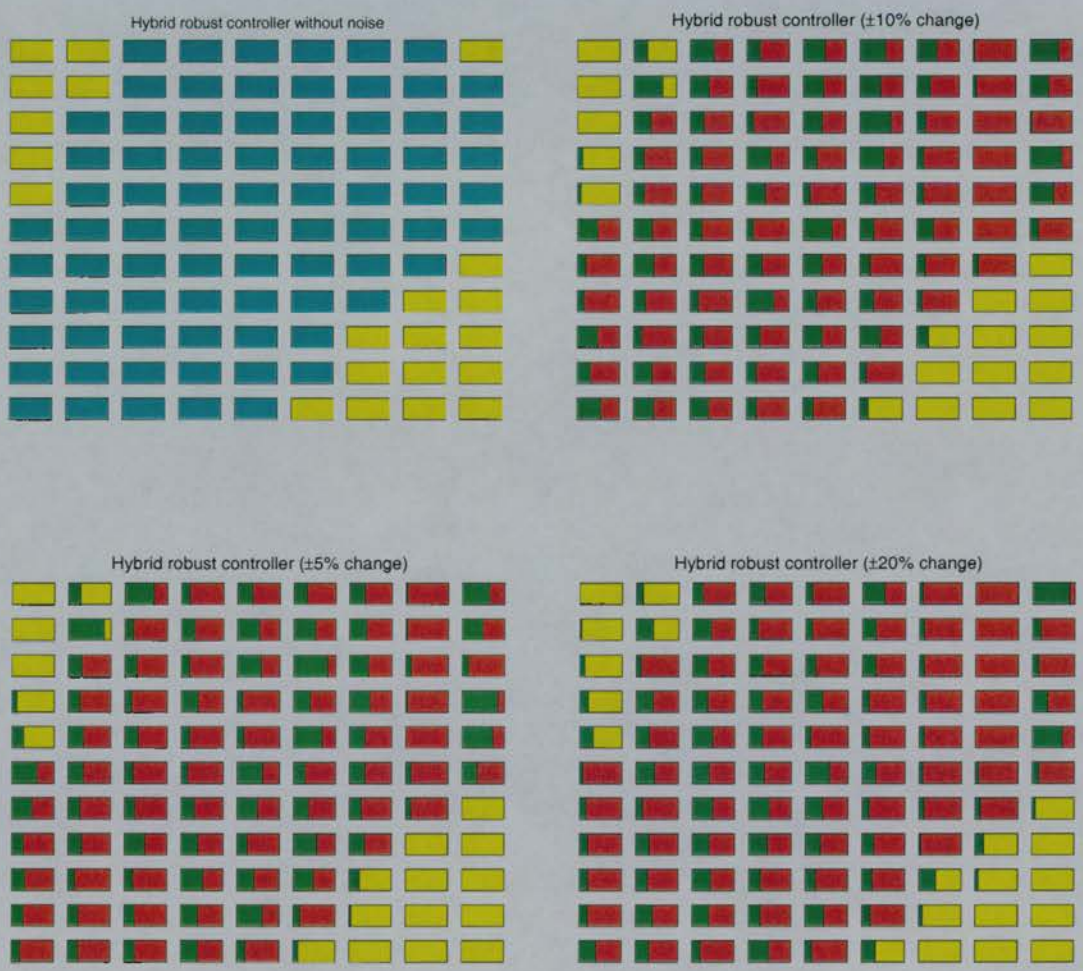


Figure F.7: Results of Analysis 2 (hybrid robust controller).

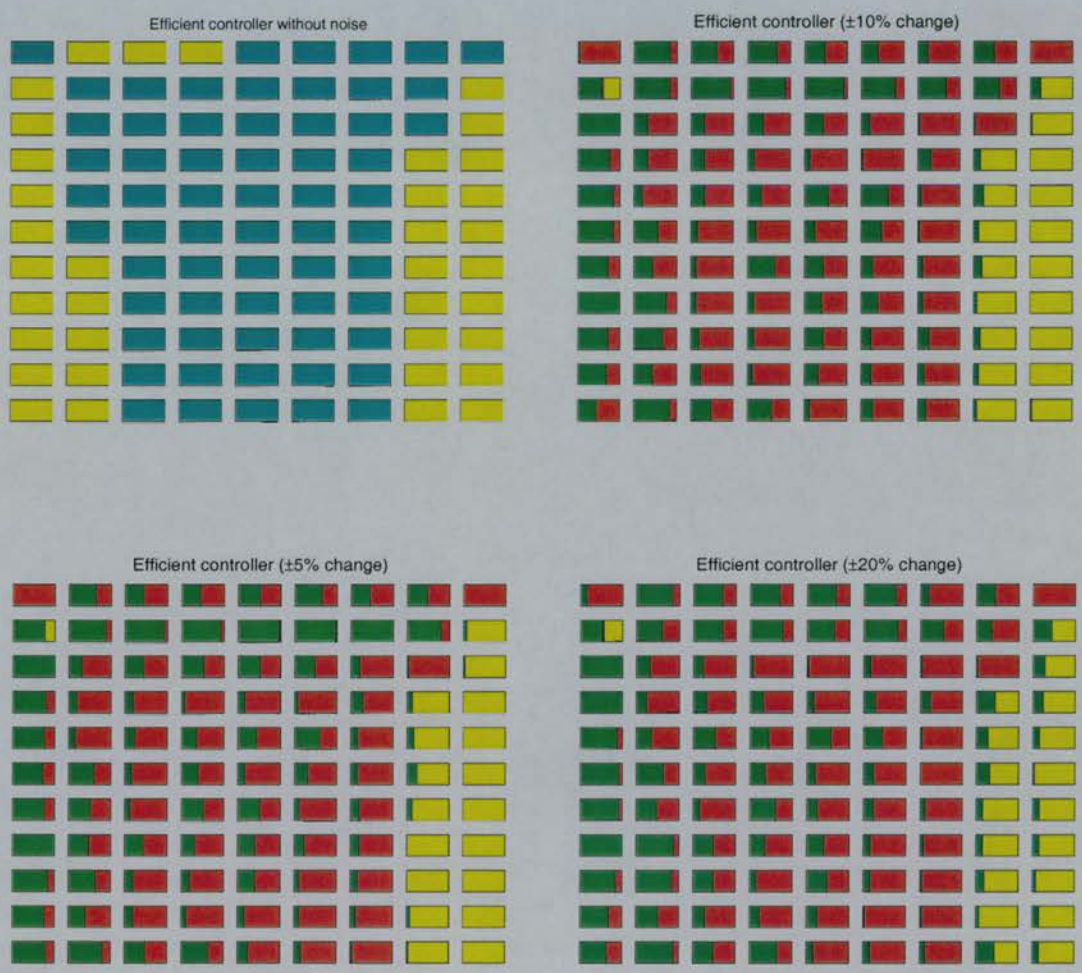


Figure F.8: Results of Analysis 2 (efficient controller).

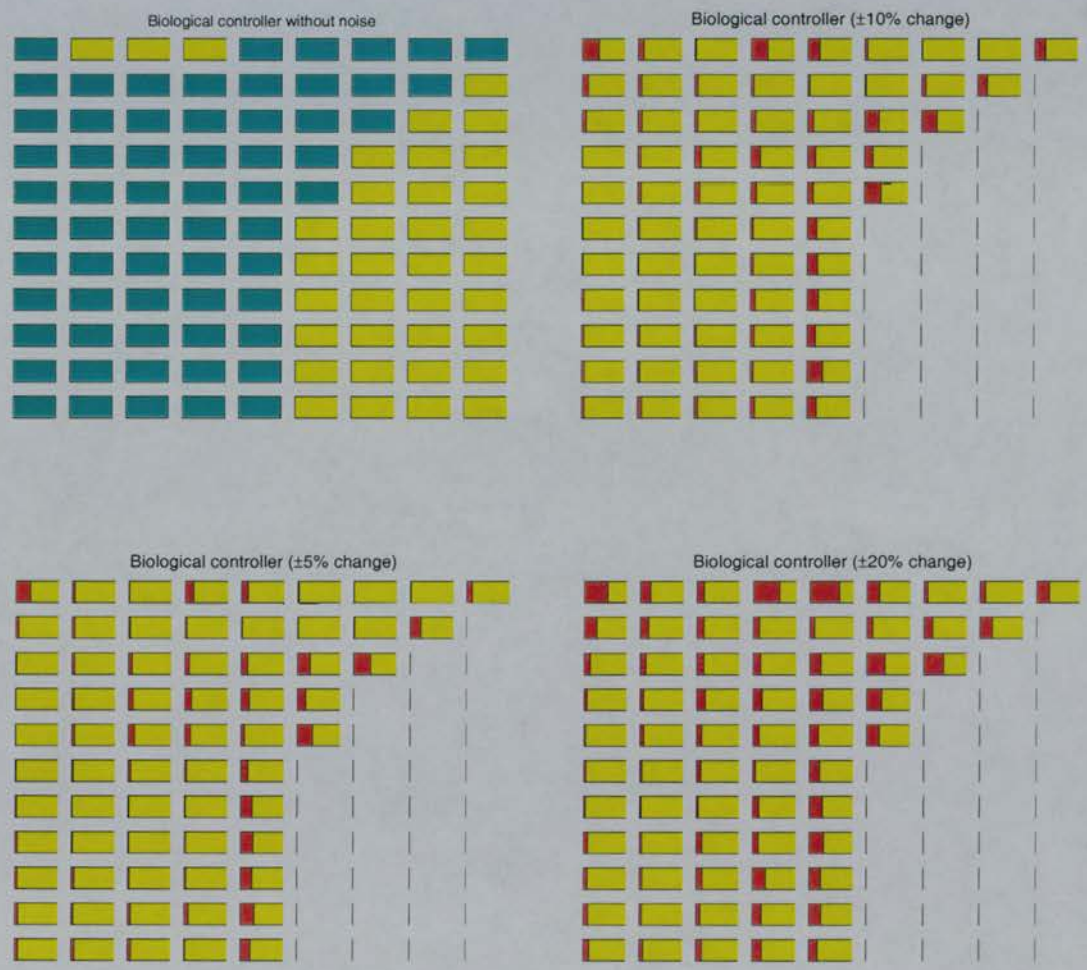


Figure F.9: Results of Analysis 3 (biological controller).

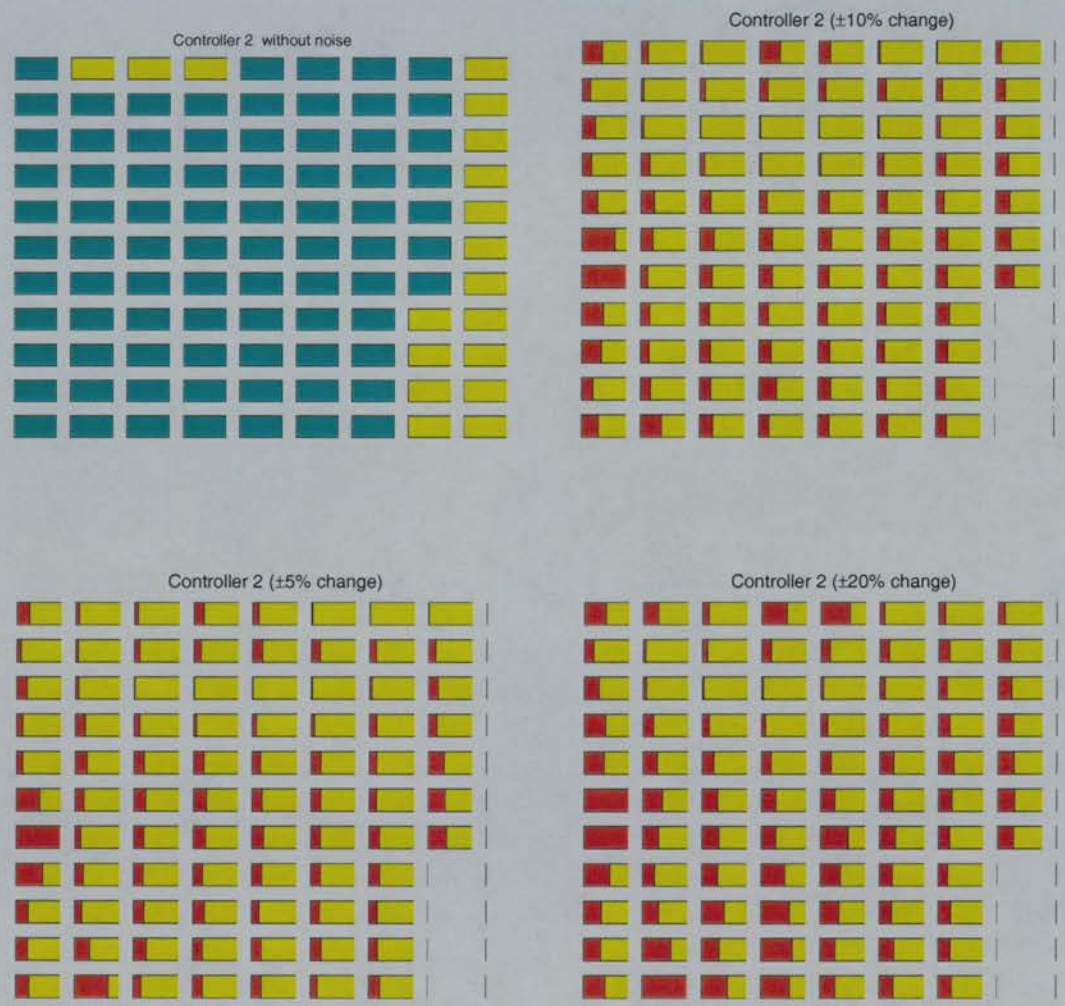


Figure F.10: Results of Analysis 3 (controller 2).

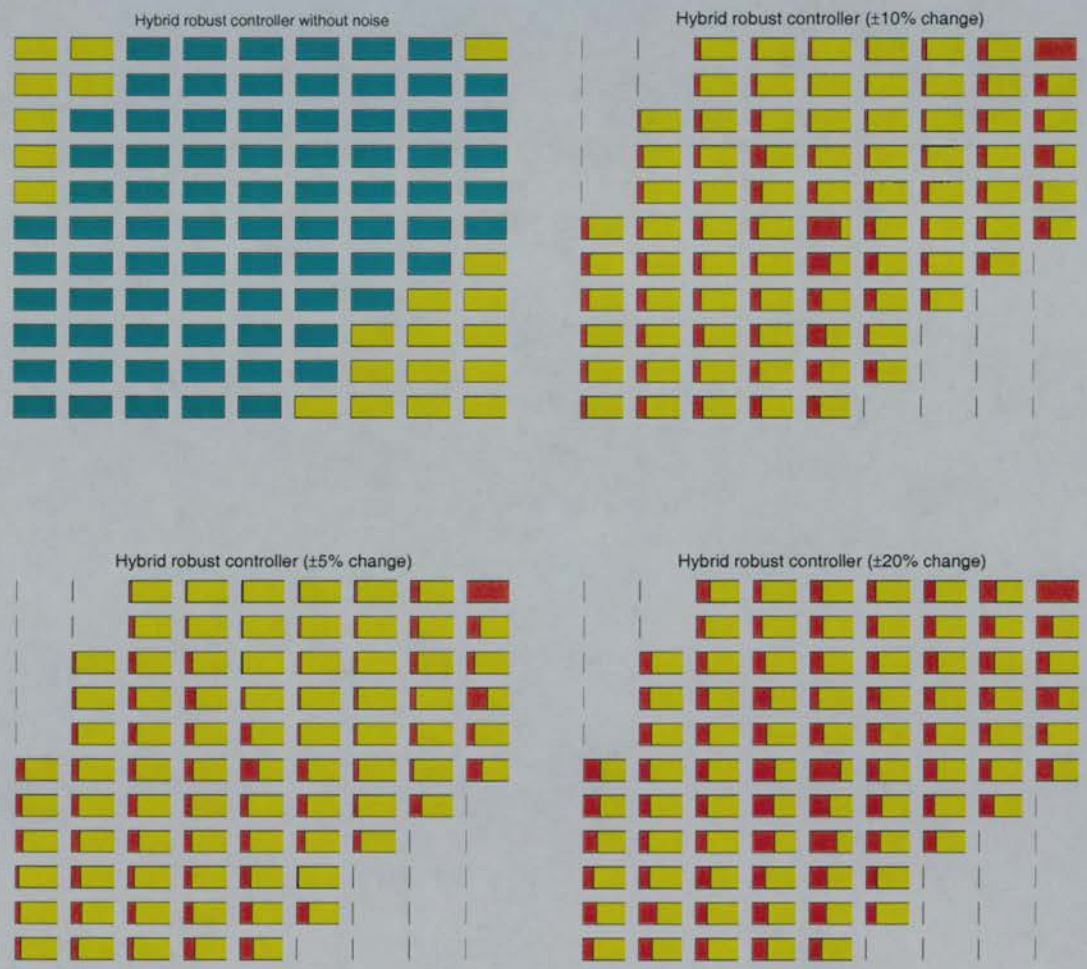


Figure F.11: Results of Analysis 3 (hybrid robust controller).

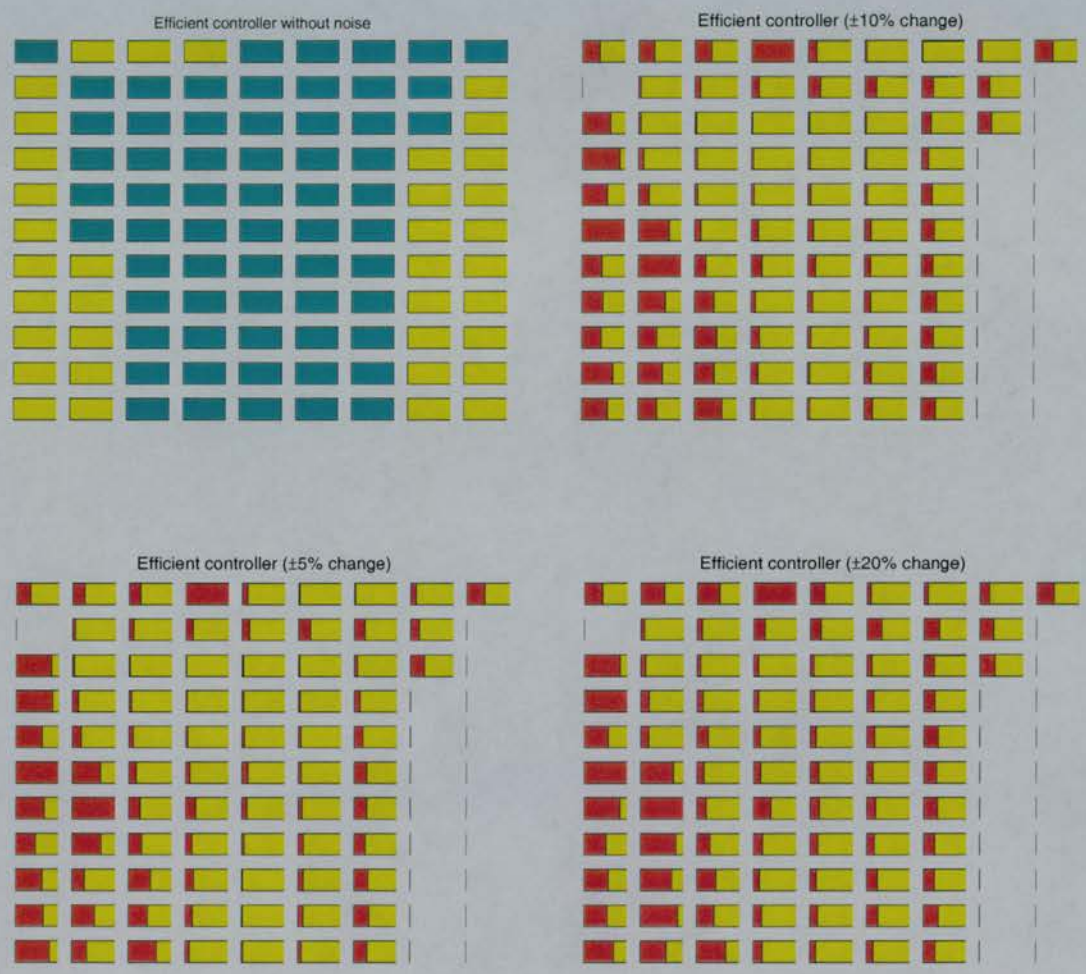


Figure F.12: Results of Analysis 3 (efficient controller).

Appendix G

Number of valid runs in all four experiments of Chapter 8

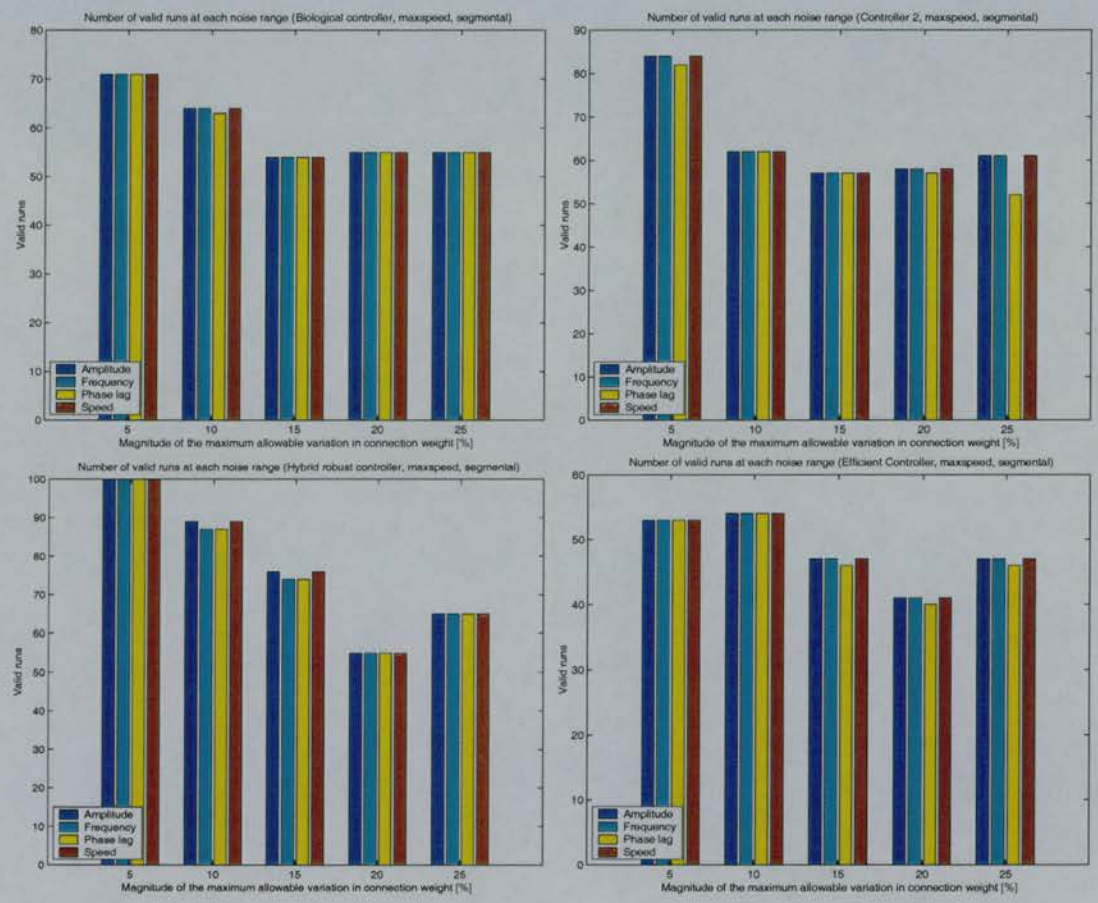


Figure G.1: Number of valid runs in each category at different noise ranges after random variation in segmental connections (Experiment 1).

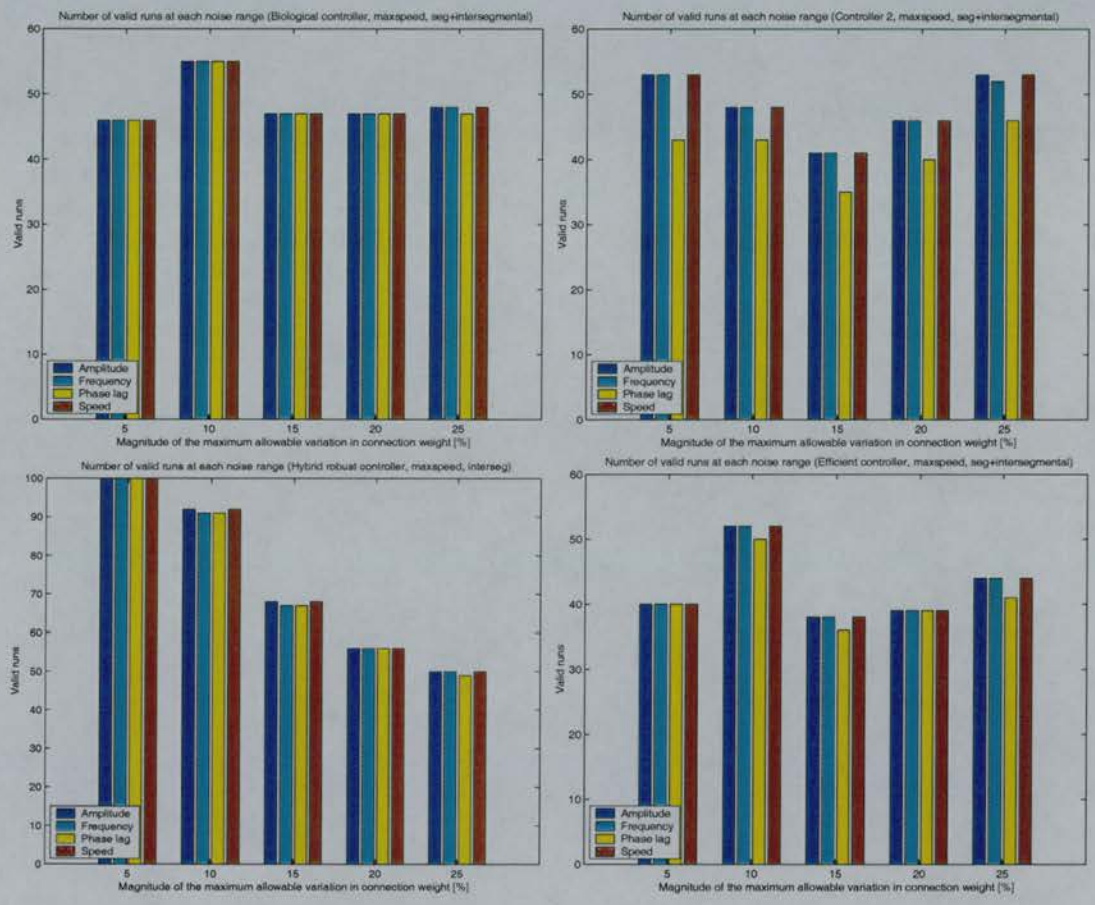


Figure G.2: Number of valid runs in each category at different noise ranges after random variation in segmental and intersegmental connections (Experiment 2).

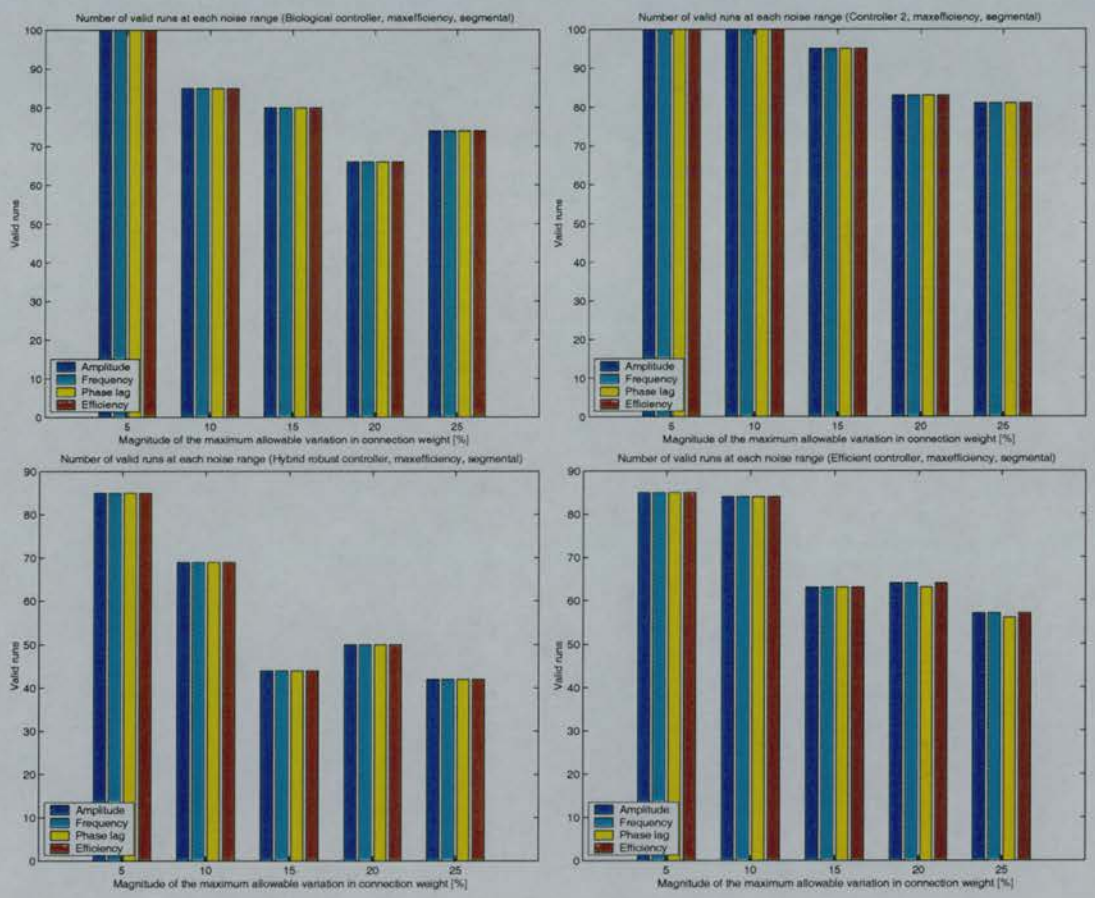


Figure G.3: Number of valid runs in each category at different noise ranges after random variation in segmental connections (Experiment 3).

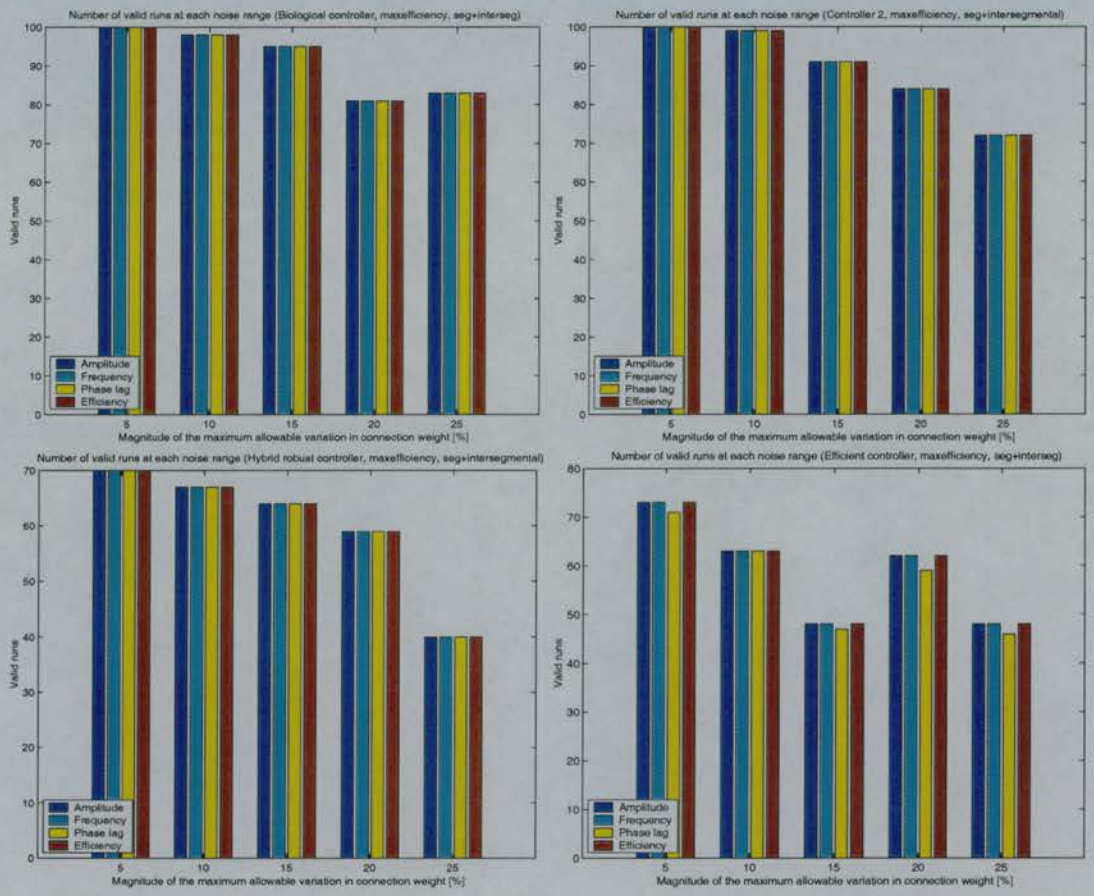


Figure G.4: Number of valid runs in each category at different noise ranges after random variation in segmental and intersegmental connections (Experiment 4).

Appendix H

Summary of performance of the biological controller, controller 2, efficient controller and hybrid robust controller

Controller	Frequency range in [Hz]	Phase lag range in [%]	Speed range in [m/s]	Efficiency range	Discrepancy	Robustness against noise	Optimization for high speed
Biological	[1.6, 5.5]	[-0.1, 1.7]	[-0.09, 0.45]	[0.05, 0.58]	0.318 [0.312]	**Very good**	Good
Controller 2	[1.7, 6.0]	**[-3.1, 3.2]**	[-0.09, 0.49]	[0.02, 0.59]	**0.223** [0.25]	Good	**Very good**
Hybrid robust	**[1.8, 7.1]**	[0.0, 3.1]	[-0.02, 0.49]	[0.08, 0.69]	0.258 [0.247]	Poor	**Very good**
Efficient	[1.6, 5.5]	[-0.3, 2.1]	**[-0.15, 0.51]**	**[0.06, 0.76]**	0.257	**Very good**	**Very good**

Table H.1: Summary of the performance of the four main types of controllers investigated in this thesis. All controllers have the same amplitude range of [0.0, 0.8]. Note that except the efficient controller (which was created after the new speed calculation algorithm was used), each controller has two discrepancy values depending on the methods used to calculate the swimming speeds. The numbers in brackets were calculated using the speeds obtained through the old speed calculation method. For discrepancy value, the smallest value is the most desirable. For other categories, largest values are better. (Note that the hybrid robust controller has the lowest discrepancy value when the speeds computed with the old method are used. However, under the new speed calculation method, controller 2 has the lowest discrepancy. Also, recall that the hybrid robust controller has the worst brainstem modulations.)

Bibliography

P. R. Adby and M. A. H. Dempster. *Introduction to Optimization Methods*. Chapman and Hall, 1974.

J. A. Anderson, editor. *An introduction to neural networks*. MIT Press, 1995.

M. A. Arbib, editor. *The handbook of brain theory and neural networks*. MIT Press, 1995.

J. Ayers. A reactive ambulatory robot architecture for operation in current and surge. In *Proceedings of the Automonous Vehicles in Mine Countermeasures Symposium*, pages 15–31. Naval Postgraduate School, 1995.

J. Ayers, J. Witting, C. Wilbur, P. Zavarcky, N. McGruer, and D. Massa. Biomimetic robots for shallow water mine countermeasures. In *Proceedings of the Automonous Vehicles in Mine Countermeasures Symposium*. Naval Postgraduate School, 2000.

J. A. Ayers, G. A. Carpenter, S. Currie, and J. Kinch. Which behavior does the lamprey central motor program mediate? *Science*, 221:1312–1314, 1983.

R. Bainbridge. The speed of swimming of fish as related to size and to the frequency and amplitude of the tail beat. *Journal of Experimental Biology*, 35:109–133, 1958.

A. G. Barto. Reinforcement learning in motor control. In M.A. Arbib, editor, *The handbook of brain theory and neural networks*, pages 809–813. MIT Press, 1995.

F. W. H. Beamish. Swimming performance of adult sea lamprey, *petromyzon marinus* in relation to weight and temperature. *Transactions of the American Fisheries Society*, 103:355–358, 1974.

E. Bizzi, F. A. Mussa-Ivaldi, and S. Giszter. Computations underlying the execution of movement: a biological perspective. *Science*, 253:287–291, 1991.

R.W. Blake. *Fish locomotion*. Cambridge University Press, 1993.

L. Bloomfield. *How things work: The physics of everyday life*. Wiley, 1997.

C. A. Breder. The locomotion of fishes. *Zoologica*, 4:159–297, 1926.

L. Brodin, J. Christenson, and S. Grillner. Single sensory neurons activate excitatory amino acid receptors in the lamprey spinal cord. *Neuroscience Letters*, 75:75–79, 1987.

J. T. Buchanan. Identification of interneurons with contralateral, caudal axons in the lamprey spinal cord: synaptic interactions and morphology. *Journal of Neurophysiology*, 47:961–976, 1982.

J. T. Buchanan. Neural network simulations of coupled locomotor oscillators in the lamprey spinal cord. *Biological Cybernetics*, 66:367–374, 1992.

J. T. Buchanan. Commissural interneurons in rhythm generation and intersegmental coupling in the lamprey spinal cord. *Journal of Neurophysiology*, 81:2037–2045, 1999.

J. T. Buchanan and S. Grillner. Newly identified 'glutamate interneurons' and their role in locomotion in the lamprey spinal cord. *Science*, 236:312–314, 1987.

R. L. Burden and J. D. Faires. *Numerical Analysis*. PWS Publishing Company, fifth edition, 1993.

J. W. Burdick and J. P. Ostrowski. The mechanics and control of undulatory locomotion. *International Journal of Robotics Research*, 117(7):683–701, 1998.

J. M. Camhi and W. Tom. The escape behavior of the cockroach *periplaneta americana*. I. Turning response to wind puffs. *Journal of Comparative Physiology*, 128: 193–201, 1978.

J. Carling, T. L. Williams, and G. Bowtell. Self-propelled anguilliform swimming: Simultaneous solution of the two-dimensional navier-stokes equations and newton's laws of motion. *Journal of Experimental Biology*, 201:3143–3166, 1998.

A. H. Cohen. Evolution of the vertebrate central pattern generator for locomotion. In A. H. Cohen, S. Rossignol, and S. Grillner, editors, *Neural control of rhythmic movements in vertebrates*. Jon Wiley & Sons, 1988.

A. H. Cohen, B. Ermentrout, T. Kiemel, N. Kopell, K. A. Sigvardt, and T. L. Williams. Modelling of intersegmental coordination in the lamprey central pattern generator for locomotion. *Trends in Neuroscience*, 15(11):434–438, 1992.

A. H. Cohen and R. M. Harris-Warrick. Strychnine eliminates alternating motor output during fictive locomotion in the lamprey. *Brain Research*, 293:164–167, 1984.

A. H. Cohen, P. J. Holmes, and R. H. Rand. The nature of the coupling between segmental oscillators of the lamprey spinal generator for locomotion: A mathematical model. *Journal of Mathematical Biology*, 13:345–369, 1982.

A. H. Cohen and P. Wallén. The neuronal correlate of locomotion in fish. 'Fictive swimming' induced in an *in vitro* preparation of the lamprey spinal cord. *Experimental Brain Research*, 41, 1980.

K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. Wiley, 2002.

F. Delcomyn. Neural basis for rhythmic behaviour in animals. *Science*, 210:492–498, 1980.

F. Delcomyn. *Foundations of Neurobiology*. W. H. Freeman and Company, 1998.

T. O. Deliagina, G. Orlovsky, S. Grillner, and P. Wallén. Vestibular control of swimming in lamprey: II. Characteristics of spatial sensitivity of reticulospinal neurons. *Experimental Brain Research*, 90:489–498, 1992a.

T. O. Deliagina, G. Orlovsky, S. Grillner, and P. Wallén. Vestibular control of swimming in lamprey: III. Activity of vestibular afferents. Convergence of vestiular inputs on reticulospinal neurons. *Experimental Brain Research*, 90:499–507, 1992b.

I. Delvolvé, T. Bem, and J.-M. Cabelguen. Epaxial and limb muscle activity during swimming and terrestrial stepping in the adult newt, *pleurodeles waltl*. *Journal of Neurophysiology*, 78:638–650, 1997.

P. S. Dickinson, C. Mecsas, and E. Marder. Neuropeptide fusion of two motor pattern generator circuits. *Nature*, 344:155–158, 1990.

K. Doya. Recurrent networks: Supervised learning. In M.A. Arbib, editor, *The handbook of brain theory and neural networks*. MIT Press, 1995.

Ö. Ekeberg. A combined neuronal and mechanical model of fish swimming. *Biological Cybernetics*, 69:363–374, 1993.

Ö. Ekeberg, A. Lansner, and S. Grillner. The neural control of fish swimming studied through numerical simulations. *Adaptive Behavior*, 3(4):363–384, 1995.

Ö. Ekeberg, P. Wallén, L. Brodin, and A. Lansner. A computer-based model for realistic simulations of neural networks I: The single neuron and synaptic interaction. *Biological Cybernetics*, 65:81–90, 1991.

J. L. Elman. Finding structure in time. *Cognitive Science*, 14:179–211, 1990.

C. H. Ferrar, T. L. Williams, and G. Bowtell. The effect of cell duplication and noise in a pattern generating network. *Neural Computation*, 5:587–596, 1993.

R. Fitzgerald. How fast could tyrannosaurus rex run? *Physics Today*, 55:18, 2002.

P. A. Getting. Understanding central pattern generators: insights gained from the study of invertebrate systems. In S. Grillner, P. S. G. Stein, G. Stuart, H. Forssberg, and R. M. Herman, editors, *Neurobiology of vertebrate locomotion*. Macmillan, 1986.

P. A. Getting. Comparative analysis of invertebrate central pattern generators. In A. H. Cohen, S. Rossignol, and S. Grillner, editors, *Neural control of rhythmic movements in vertebrates*. Jon Wiley & Sons, 1988.

P. A. Getting. Reconstruction of small neural networks. In C. Koch and I. Segev, editors, *Methods in neural modeling*, pages 171–196. MIT Press, 1989.

P. A. Getting and M. S. Dekin. Tritonia swimming: A model system for integration within rhymthic motor system. In A. I. Selverston, editor, *Model neural networks and behavior*, pages 3–20. Plenum, 1985.

D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, 1989.

R. M. Golden. *Mathematical methods for neural network: analysis and design*. MIT-Press, 1996.

S. Grillner. On the generation of locomotion in the spinal dogfish. *Brain Research*, 20: 459–470, 1974.

S. Grillner. Control of locomotion in bipeds, tetrapods, and fish. In V. B. Brooks, editor, *Handbook of Physiology. Section 1. The Nervous system II. Motor Control*, pages 1179–1236. American Physiological Society, 1981.

S. Grillner. Neural control of vetebrate locomotion – central mechanisms and reflex interaction with special reference to the cat. In W. J. P. Barnes and Gladden M. H., editors, *Feedback and motor control in invertebrates and vertebrates*, pages 35–56. Croom Helm, 1985.

S. Grillner. Neural networks for vertebrate locomotion. *Scientific American*, pages 48–53, January 1996.

S. Grillner, J. T. Buchanan, P. Wallén, and L. Brodin. Neural control of locomotion in lower vertebrates. In A. H. Cohen, S. Rossignol, and S. Grillner, editors, *Neural control of rhythmic movements in vertebrates*, pages 1–40. Jon Wiley & Sons, 1988.

S. Grillner, T. Degliana, Ö. Ekeberg, A. El Marina, A. Lansner, G. N. Orlovsky, and P. Wallén. Neural networks that co-ordinate locomotion and body orientation in lamprey. *Trends in Neuroscience*, 18(6):270–279, 1995.

S. Grillner and S. Kashin. On the generation and performance of swimming in fish. In A. H. Cohen, S. Rossignol, and S. Grillner, editors, *Neural control of locomotion*. Plenum Press, 1976.

S. Grillner, C. Perret, and P. Zangger. Central generation of locomotion in the spinal dogfish. *Brain Research*, 109:225–269, 1976.

S. Grillner and P. Wallén. How does the lamprey central nervous system make the lamprey swim? *Journal of Experimental Biology*, 112:337–357, 1984.

S. Grillner, P. Wallén, and L. Brodin. Neuronal network generating locomotor behavior in lamprey: Circuitry, transmitters, membrane properties, and simulation. *Annual Review of Neuroscience*, 14:169–199, 1991a.

S. Grillner, P. Wallén, A. McClellan, K. Sigvardt, T. Williams, and J. Feldman. The neural generation of locomotion in the lamprey: an incomplete account. In A. Roberts and B. Roberts, editors, *Neural origin of rhythmic movements*, pages 285–303. Cambridge University Press, 1983.

S. Grillner, P. Wallén, and G. Viana di Prisco. The lamprey locomotor system - Basic locomotor synergy. In M. Shimamura, S. Grillner, and V. R. Edgerton, editors, *Neurobiological basis of human locomotion*, pages 77–92. Japan Scientific Societies Press, 1991b.

R. M. Harris-Warrick. Chemical modulation of central pattern generators. In A. H. Cohen, S. Rossignol, and S. Grillner, editors, *Neural control of rhythmic movements in vertebrates*, pages 1–40. Jon Wiley & Sons, 1988.

R. M. Harris-Warrick and A. H. Cohen. Serotonin modulates the central pattern generator for locomotion in the isolated lamprey spinal cord. *Journal of Experimental Biology*, 116:27–46, 1985.

R. M. Harris-Warrick and E. Marder. Modulation of neural networks for behavior. *Annual Review of Neuroscience*, 14:39–57, 1991.

J. Hellgren, S. Grillner, and A. Lansner. Computer simulation of the segmental neural network generating locomotion in lamprey by using populations of network interneurons. *Biological Cybernetics*, 68:1–13, 1992.

J. Hellgren, S. Grillner, and A. Lansner. Neural mechanisms potentially contributing to the intersegmental phase lag in lamprey. i: Segmental oscillations dependent on reciprocal inhibition. *Biological Cybernetics*, 81:317–330, 1999a.

J. Hellgren, A. Lansner, and G. Grillner. Neural mechanisms potentially contributing to the intersegmental phase lag in lamprey. ii: Hemisegmental oscillations produced by mutally coupled excitatory neurons. *Biological Cybernetics*, 81:299–315, 1999b.

J. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, 1975.

M. E. J. Holwill. Low Reynolds number undulatory propulsion in organisms of different sizes. In T. J. Pedley, editor, *Scale effects in animal locomotion*, pages 233–242. Academic Press, 1977.

S. L. Hopper and M. Moulins. Switching of a neuron from one network to another by sensory-induced changes in membrane properties. *Science*, 244:1587–1589, 1989.

A. J. Ijspeert. *Design of artificial neural oscillatory circuits for the control of lamprey- and salamander-like locomotion using evolutionary algorithms*. PhD thesis, University of Edinburgh, 1998.

A. J. Ijspeert. A connectionist central pattern genertor for the aquatic and terrestrial gaits of a simulated salamander. *Biological Cybernetics*, 84(5):331–348, 2001.

A. J. Ijspeert, J. Hallam, and D. Willshaw. Evolving swimming controllers for a simulated lamprey with inspiration from neurobiology. *Adaptive Behavior*, 7(2):151–172, 1999.

N. Jakobi. The minimal simulation approach to evolutionary robotics. In T. Gomi, editor, *Evolutionary Robotics*. AAI Books, 1998.

N. Jakobi, P. Husbands, and I. Harvey. Noise and the reality gap: the use of simulation in evolutionary robotics. In F.Moran, A.Moreno, J.J. Merelo, and P.Chacon, editors, *Proceedings of the third European Conference on Artificial Life, ECAL95*. Springer-Verlag, 1995.

M. I. Jordan. Attractor dynamics and parallelism in a connectionist sequential machine. In *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*, pages 531–546. Lawrence Erlbaum Associates, 1986.

R. Jung, T. Kiemel, and A. H. Cohen. Dynamical behavior of a neural network model of locomotor control in the lamprey. *Journal of Neurophysiology*, 75:1074–1086, 1996.

N. Kato and T. Inaba. Guidance and control of fish robot with apparatus of pectoral fin motion. In *Proceedings of IEEE International Conference on Robotics and Automation.*, pages 446–451. IEEE Press, 1998.

D. Kleinfeld and H. Sompolinsky. Associative network models for central pattern generators. In C. Koch and I. Segev, editors, *Methods in neural modeling*, pages 195–246. MIT Press, 1989.

T. Knutsen. Designing an underwater eel-like robot and developing anguilliform locomotion control, 2000. `http://www.ee.upenn.edu/~sunfest/pastProjects/Papers00/KnutsenTamara.pdf`.

N. Kopell, G. B. Ermentrout, and T. L. Williams. On chains of oscillators forced at one end. *SIAM, Journal of Applied Mathematics*, 51(5):1397–1417, 1991.

J. Koza. *Genetic Programming*. MIT Press, 1992.

W.B. Langdon and R. Poli. *Foundations of Genetic Programming*. Springer Verlag, 2002.

A. Lansner, Ö. Ekeberg, and S. Grillner. Realistic modeling of burst generation and swimming in lamprey. In P. Stein, S. Grillner, A. I. Selverston, and D. Stuart, editors, *Neurons, Networks, and Motor Behavior*, pages 165–171. MIT Press, 1997.

M. J. Lighthill. Note on the swimming of slender fish. *Journal of Fluid Mechanics*, 9: 305–317, 1960.

M. J. Lighthill. Aquatic animal propulsion of high hydrodynamic efficiency. *Journal of Fluid Mechanics*, 44:263–301, 1970a.

M. J. Lighthill. How do fishes swim? *Endeavour*, 29:77–83, 1970b.

E. Marder, N. Kopell, and K. A. Sigvardt. How computation aids in understanding biological networks. In P. Stein, S. Grillner, A. I. Selverston, and D. Stuart, editors, *Neurons, Networks, and Motor Behavior*, pages 139–149. MIT Press, 1997.

C. D. Marsden, J. C. Rothwell, and B. L. Day. The use of peripheral feedback in the control of movement. *Trends in Neuroscience*, 7:253–258, 1984.

T. Matsushima and S. Grillner. Neural mechanisms of intersegmental coordination in lamprey: local excitability changes modify the phase coupling along the spinal cord. *Journal of Neurophysiology*, 67:373–388, 1992.

K. A. McIsaac and J. P. Ostrowski. A geometric approach to anguilliform locomotion: simulation and experiments with an underwater eel robot. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 1, pages 2843–2848. IEEE Press, 1999.

W. L. Miller and K. A. Sigvardt. Extent and role of multisegmental coupling in the lamprey spinal locomotor pattern generator. *Journal of Neurophysiology*, 83:465–476, 2000.

J. A. Nelder and R. Mead. A simplex method for function minimization. *Computer Journal*, 7:308–313, 1965.

S. Nolfi and D. Floreano. *Evolutionary Robotics: the biology, intelligence, and technology of self-organizing machines*. MIT Press, 2000.

J. Or and J. Hallam. A study on the robustness of the lamprey swimming controllers. In J. A. Meyer, A. Berthoz, D. Floreano, H. L. Roitblat, and S. W. Wilson, editors, *SAB2000 Proceedings Supplement, Sixth International Conference on Simulation of Adaptive Behaviour*, pages 68–76. MIT Press, 2000.

J. Or, J. Hallam, D. Willshaw, and A. Ijspeert. Evolution of robust swimming controllers for a simulated lamprey, 2002. unpublished work.

G. Orlovsky, T. Deliagina, and P. Wallén. Vestibular control of swimming in lamprey: I. Responses of reticulospinal neurons to roll and pitch. *Experimental Brain Research*, 90:479–488, 1992.

K. G. Pearson. Common principles of motor control in vertebrates and invertebrates. *Annual Review of Neuroscience*, 16:265–297, 1993.

K. Petersen. *Vertebrate natural history*. Zoology 451, Lecture notes. University of Washington, 2001.

M. L. T. Poon. Induction of swimming in lamprey by l-dopa and amino acids. *Journal of Comparative Physiology*, 136:337–344, 1980.

R. H. Rand, A. H. Cohen, and P. J. Holmes. Systems of coupled oscillators as models of central pattern generators. In A. H. Cohen, S. Rossignol, and S. Grillner, editors, *Neural control of rhythmic movements in vertebrates*. Jon Wiley & Sons, 1988.

R. E. Ritzmann and A. J. Pollack. Parallel motor pathways from thoracic interneurons of the ventral giant interneuron system of the cockroach, *periplaneta americana*. *Journal of Neurobiology*, 21:1219–1235, 1990.

P. Ross, J. Hallam, and C. Williams. *Connectionist Computing*. Department of Artificial Intelligence, University of Edinburgh, 1998.

C. M. Rovainen. Synaptic interactions of identified nerve cells in the spinal cord of the sea lamprey. *Journal of Comparative Neurology*, 154:189–206, 1974.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

K. Schmidt-Nielsen. *Scaling: Why is animal size so important?* Cambridge University Press, 1984.

R. A. Serway, R. J. Beichner, and J. W. Jewett. *Physics for scientists and engineers with modern physics*. Saunders College Publishing, fifth edition, 2000.

M. Sfakiotakis, D. Lane, and J. Davis. Review of fish swimming modes for aquatic locomotions. *IEEE Journal of Oceanic Engineering*, 24(2):237–252, 1999.

K. A. Sigvardt and T. L. Williams. Effects of local oscillator frequency on intersegmental coordination in the lamprey locomotor CPG: theory and experiment. *Journal of Neurophysiology*, 76(6):4094–4103, December 1996.

M. Ullström, J. Hellgren Kotaleski, J. Tegnér, E. Aurell, S. Grillner, and A. Lansner. Activity dependent modulation of adaptation produces a constant burst proportion in a model of the lamprey spinal locomotor generator. *Biological Cybernetics*, 79: 1–14, 1998.

J. J. Videler. *Fish swimming*. Chapman and Hall, 1993.

T. Wadden, J. Hellgren, A. Lansner, and S. Grillner. Intersegmental coordination in the lamprey: simulations using a network model without segmental boundaries. *Biological Cybernetics*, 76:1–9, 1997.

P. Wallén, Ö. Ekeberg, A. Lansner, L. Brodin, H. Traven, and S. Grillner. A computer-based model for realistic simulations of neural networks II: The segmental network generating locomotor rhythmicity in the lamprey. *Journal of Neurophysiology*, 68: 1939–1950, December 1992.

P. Wallén, P. Grafe, and S. Grillner. Phasic variations of extracellular potassium during fictive swimming in the lamprey spinal cord in vitro. *Acta Physiologica Scandinavica*, 120:457–463, 1984.

P. Wallén and T. Willams. Fictive locomotion in the lamprey spinal cord in vitro compared with swimming in the intact and spinal animal. *Journal of Physiology*, 347: 225–239, 1984.

C. S. Wardle. Effects of size on the swimming speeds of fish. In T. J. Pedley, editor, *Scale effects in animal locomotion*, pages 299–313. Academic Press, 1977.

P. W. Webb. Effects of size on performance and energetics of fish. In T. J. Pedley, editor, *Scale effects in animal locomotion*, pages 315–329. Academic Press, 1977.

P. W. Webb and P. T. Kostecki. The effect of size and swimming speed on locomotor kinematics of rainbow trout. *Journal of Experimental Biology*, 109:77–95, 1984.

D. R. Wilkie. Metabolism and body size. In T. J. Pedley, editor, *Scale effects in animal locomotion*, pages 23–36. Academic Press, 1977.

T. L. Williams. Mechnical and neural patterns underlying swimming by lateral undulations: Review of studies on fish, amphibia and lamprey. In S. Grillner, P. S. G. Stein, D. Stuart, H. Forssberg, and R. Herman, editors, *Neurobiology of vertebrate locomotion*, pages 141–155. Macmillan, 1986.

T. L. Williams. Phase coupling by synaptic spread in chains of coupled neuronal oscillators. *Science*, 258:662–665, 1992a.

T. L. Williams. Phase coupling in simulated chains of coupled neuronal oscillators representing the lamprey spinal cord. *Neural Computation*, 4:546–558, 1992b.

T. L. Williams and K. A. Sigvardt. Spinal cord of lamprey: generation of locomotor patterns. In M.A. Arbib, editor, *The handbook of brain theory and neural networks*, pages 918–921. MIT Press, 1995.

T. L. Williams, K. A. Sigvardt, N. Kopell, G. B. Ermentrout, and M. P. Rempler. Forcing of coupled nonlinear oscillators: studies of intersegmental coordination in the lamprey locomotor central pattern generator. *Journal of Neurophysiology*, 64:862–871, September 1990.

T. Y. Wu. Introduction to the scaling of aquatic animal locomotion. In T. J. Pedley, editor, *Scale effects in animal locomotion*, pages 203–232. Academic Press, 1977.

P. Yam and J. Or. Roaches on the wheels. *Scientific American*, pages 48–49, January 1998.